



DEVELOPER GUIDE FOXIT PDF SDK

For C API

TABLE OF CONTENTS

1. Introduction to Foxit PDF SDK	1
1.1 Why Choose Foxit PDF SDK.....	1
1.2 Foxit PDF SDK for C API.....	2
1.3 Evaluation	2
1.4 License	2
1.5 About this guide.....	2
2. Getting Started.....	3
2.1 System Requirements.....	3
2.2 What is in the package	3
2.3 How to run a demo	4
2.4 How to create a simple project.....	6
3. Working with SDK API.....	12
3.1 Initialize Library	12
3.1.1 How to initialize Foxit PDF SDK.....	12
3.2 Document.....	12
3.2.1 How to create a PDF document from scratch.....	13
3.2.2 How to load an existing PDF document from file path	13
3.2.3 How to load an existing PDF document from a memory buffer	13
3.2.4 How to load an existing PDF document from a file read callback object.....	13
3.2.5 How to load PDF document and get the first page of the PDF document.....	15
3.2.6 How to save a PDF to a file.....	15
3.2.7 How to save a document into memory buffer by WriterCallback.....	15
3.3 Page.....	16
3.3.1 How to get page size.....	16
3.3.2 How to calculate bounding box of page contents.....	17
3.3.3 How to create a PDF page and set the size.....	17

3.3.4	How to delete a PDF page	17
3.3.5	How to flatten a PDF page.....	17
3.3.6	How to get and set page thumbnails in a PDF document.....	18
3.4	Render.....	18
3.4.1	How to render a page to a bitmap	19
3.4.2	How to render page and annotation	20
3.5	Attachment.....	20
3.5.1	How to export the embedded attachment file from a PDF and save it as a single file	20
3.5.2	How to remove all the attachments of a PDF.....	21
3.6	Text Page	22
3.6.1	How to extract text from a PDF page.....	22
3.6.2	M	23
3.6.3	How to get the text within a rectangle area in a PDF.....	23
3.7	Text Search.....	23
3.7.1	How to search a text pattern in a PDF.....	23
3.8	Search and Replace	24
3.8.1	System requirements	24
3.8.2	How to work with the search and replace function	25
3.9	Text Link.....	25
3.9.1	How to retrieve hyperlinks in a PDF page	25
3.10	Bookmark	26
3.10.1	How to find and list all bookmarks of a PDF.....	26
3.10.2	How to insert a new bookmark	27
3.10.3	How to create a table of contents based on bookmark information in PDFs	28
3.11	Form (AcroForm)	29
3.11.1	How to load the forms in a PDF.....	29
3.11.2	How to count form fields and get/set the properties.....	29
3.11.3	How to export the form data in a PDF to a XML file	30

3.11.4	How to import form data from a XML file	30
3.11.5	How to get coordinates of a form field.....	31
3.12	XFA Form.....	32
3.12.1	How to load XFADoc and represent an Interactive XFA form	32
3.12.2	How to export and import XFA form data.....	33
3.13	Form Design	34
3.13.1	How to add a text form field to a PDF	34
3.13.2	How to remove a text form field from a PDF.....	35
3.14	Annotations	35
3.14.1	General	35
3.14.2	Import annotations from or export annotations to a FDF file.....	42
3.15	Image Conversion.....	43
3.15.1	How to convert PDF pages to bitmap files	43
3.15.2	How to convert an image file to PDF file	45
3.16	Watermark.....	46
3.16.1	How to create a text watermark and insert it into the first page	46
3.16.2	How to create an image watermark and insert it into the first page.....	47
3.16.3	How to remove all watermarks from a page	48
3.17	Barcode.....	48
3.17.1	How to generate a barcode bitmap from a string	48
3.18	Security	49
3.18.1	How to encrypt a PDF file with Certificate.....	49
3.18.2	How to encrypt a PDF file with Foxit DRM.....	50
3.19	Reflow	51
3.19.1	How to create a reflow page and render it to a bmp file	51
3.20	Asynchronous PDF	52
3.21	Pressure Sensitive Ink.....	53
3.21.1	How to create a PSI and set the related properties for it.....	53

3.22	Wrapper	53
3.22.1	How to open a document including wrapper data	54
3.23	PDF Objects	54
3.23.1	How to remove some properties from catalog dictionary	54
3.24	Page Object	55
3.24.1	How to create a text object in a PDF page	55
3.24.2	How to add an image logo to a PDF page	56
3.25	Marked content	57
3.25.1	How to get marked content in a page and get the tag name	57
3.26	Layer	58
3.26.1	How to create a PDF layer	58
3.26.2	How to set all the layer nodes information	59
3.26.3	How to edit layer tree	60
3.27	Signature	60
3.27.1	How to sign the PDF document with a signature	61
3.27.2	How to implement signature callback function of signing	62
3.28	Long term validation (LTV)	73
3.28.1	How to establish long term validation of signatures using the default signature callback for subfilter "ETSI.RFC3161" and the default RevocationCallback	73
3.29	PAdES	75
3.30	PDF Action	76
3.30.1	How to create a URI action and insert to a link annot	76
3.30.2	How to create a GoTo action and insert to a link annot	77
3.31	JavaScript	77
3.31.1	How to add JavaScript Action to Document	78
3.31.2	How to add JavaScript Action to Annotation	78
3.31.3	How to add JavaScript Action to FormField	78
3.31.4	How to add a new annotation to PDF using JavaScript	80

3.31.5	How to get/set properties of annotations (strokeColor, fillColor, readOnly, rect, type) using JavaScript	80
3.31.6	How to destroy annotation using JavaScript.....	81
3.32	Redaction.....	81
3.32.1	How to redact the text "PDF" on the first page of a PDF.....	82
3.33	Comparison.....	84
3.33.1	How to compare two PDF documents and save the differences between them into a PDF file	84
3.34	OCR.....	86
3.34.1	System requirements	86
3.34.2	Trial limit for SDK OCR add-on module	86
3.34.3	OCR resource files.....	87
3.34.4	How to run the OCR demo.....	87
3.35	Compliance.....	89
3.35.1	System requirements	91
3.35.2	Compliance resource files.....	91
3.35.3	How to run the compliance or preflight demo.....	91
3.36	Optimization.....	94
3.36.1	How to optimize PDF files by compressing the color/grayscale/monochrome images	95
3.37	HTML to PDF Conversion.....	95
3.37.1	System requirements	96
3.37.2	HTML to PDF engine files	96
3.37.3	How to run the html2pdf demo	96
3.37.4	How to work with Html2PDF API	100
3.37.5	How to get HTML data from stream and convert it to a PDF file	100
3.38	Office to PDF Conversion with third-party engines	102
3.38.1	System requirements	103
3.38.2	How to convert Word to PDF.....	103
3.38.3	How to convert Excel to PDF.....	103

3.38.4	How to convert PowerPoint to PDF.....	104
3.39	Office to PDF Conversion without third-party engines	104
3.39.1	System requirements	104
3.39.2	Office to PDF resource files (Foxit PDF Conversion SDK)	105
3.39.3	How to run the office2pdf demo using Foxit PDF Conversion SDK	105
3.39.4	How to convert office files to PDF without third-party engines	105
3.40	Output Preview	106
3.40.1	System requirements	106
3.40.2	How to run the output preview demo.....	106
3.40.3	How to do output preview using Foxit PDF SDK.....	106
3.41	Combination.....	107
3.41.1	How to combine several PDF files into one PDF file	107
3.42	PDF Portfolio	108
3.42.1	System requirements	108
3.42.2	How to create a new and blank PDF portfolio	109
3.42.3	How to create a Portfolio object from a PDF portfolio	109
3.42.4	How to get portfolio nodes	110
3.42.5	How to add file node or folder node	112
3.42.6	How to remove a node	113
3.43	Table Maker.....	113
3.43.1	System requirements	113
3.43.2	How to add table to a PDF document.....	114
3.44	Accessibility	115
3.44.1	System requirements	115
3.44.2	How to tag a PDF document	116
3.45	PDF to Office Conversion	116
3.45.1	System requirements	116
3.45.2	PDF to Office resource files.....	117
3.45.3	How to run the pdf2office demo	117

3.45.4	How to work with PDF2office API.....	119
3.46	DWG to PDF Conversion	121
3.46.1	System requirements	121
3.46.2	DWG To PDF engine files.....	121
3.46.3	How to run the dwg2pdf demo	121
3.46.4	How to convert DWG to PDF.....	121
3.47	OFD.....	122
3.47.1	System requirements	122
3.47.2	OFD engine file	122
3.47.3	How to run the ofd demo.....	123
3.47.4	How to implement the conversion between OFD file and PDF file.....	123
3.47.5	How to render OFD page	123
3.48	Paragraph Editing.....	124
3.48.1	System requirements	125
3.48.2	How to work with paragraph editing	125
3.49	3D Rendering	127
3.49.1	System requirements	127
3.49.2	How to display the 3D annotation	128
3.49.3	How to set render mode and controller.....	128
FAQ	129
Appendix	133
Supported JavaScript List	133
References	147
Support	148

1. INTRODUCTION TO FOXIT PDF SDK

Have you ever thought about building your own application that can do everything you want with PDF files? If your answer is "Yes", congratulations! You just found the best solution in the industry that allows you to build stable, secure, efficient and full-featured PDF applications.

Foxit PDF SDK provides high-performance libraries to help any software developer add robust PDF functionality to their enterprise, mobile and cloud applications across all platforms (includes Windows, Mac, Linux, Web, Android, iOS, and UWP), using the most popular development languages and environments.

1.1 Why Choose Foxit PDF SDK

Foxit is a leading software provider of solutions for reading, editing, creating, organizing, and securing PDF documents. Foxit PDF SDK libraries have been used in many of today's leading apps, and are proven, robust, and battle-tested to provide the quality, performance, and features that the industry's largest apps demand. Customers choose Foxit PDF SDK product for the following reasons:

- **Easy to integrate**

Developers can seamlessly integrate Foxit PDF SDK into their own applications.

- **Lightweight footprint**

Does not exhaust system resource and deploys quickly.

- **Cross-platform support**

Support current mainstream platforms, such as Windows, Mac, Linux, Web, Android, iOS, and UWP.

- **Powered by Foxit's high fidelity rendering PDF engine**

The core technology of the SDK is based on Foxit's PDF engine, which is trusted by a large number of the world's largest and well-known companies. Foxit's powerful engine makes the app fast on parsing, rendering, and makes document viewing consistent on a variety of devices.

- **Premium World-side Support**

Foxit offers premium support for its developer products because when you are developing mission critical products you need the best support. Foxit has one of the PDF industry's largest team of support engineers. Updates are released on a regular basis to improve user experience by adding new features and enhancements.

1.2 Foxit PDF SDK for C API

Application developers who use Foxit PDF SDK can leverage Foxit's powerful, standard-compliant PDF technology to securely display, create, edit, annotate, format, organize, print, share, secure, search documents as well as to fill PDF forms. Additionally, Foxit PDF SDK (for C++ and .NET) includes a built-in, embeddable PDF Viewer, making the development process easier and faster. For more detailed information, please visit the website <https://developers.foxitsoftware.com/pdf-sdk/>.

In this guide, we focus on the introduction of Foxit PDF SDK for C API on Windows platform.

Foxit PDF SDK for C API ships with simple-to-use APIs that can help C developers seamlessly integrate powerful PDF technology into their own projects on Windows platform. It provides rich features on PDF documents, such as PDF viewing, bookmark navigating, text selecting/copying/searching, PDF signatures, PDF forms, rights management, PDF annotations, and full text search.

1.3 Evaluation

Foxit PDF SDK allows users to download a trial version to evaluate the SDK. The trial version has no difference from a standard version except for the 10-day limitation trial period and the trail watermarks that will be generated on the PDF pages. After the evaluation period expires, customers should contact Foxit sales team and purchase licenses to continue using Foxit PDF SDK.

1.4 License

Developers should purchase licenses to use Foxit PDF SDK in their solutions. Licenses grant users permissions to release their applications based on PDF SDK libraries. However, users are prohibited to distribute any documents, sample codes, or source codes in the SDK released package to any third party without the permission from Foxit Software Incorporated.

1.5 About this guide

This guide is intended for developers who need to integrate Foxit PDF SDK with the C program language into their own applications. It aims at introducing the installation package, and the usage of SDK.

2. GETTING STARTED

It's very easy to setup Foxit PDF SDK and see it in action! This guide will provide you with a brief introduction about our SDK package. The following sections introduce the contents of system requirements, the installation package as well as how to run a demo, and create your own project.

2.1 System Requirements

Windows Vista, 7, 8 and 10 (32-bit and 64-bit)

Windows Server 2003, 2008 and 2012 (32-bit and 64-bit)

The release package includes a 32 bit version and native 64 bit version DLL library for Windows 32/64.

Note:

- *It only supports for Windows 8/10 classic style, but not for Store App or Universal App.*
- *If you are using an older version of Windows (for example, Windows 7 and Windows Server 2008), you may need to visit this link <https://support.microsoft.com/en-us/help/4019990/update-for-the-d3dcompiler-47-dll-component-on-windows> to download and install the "D3DCOMPILER_47.dll". If you do not do this, you may encounter errors.*

2.2 What is in the package

Download the Foxit PDF SDK zip for Windows (C API) package and extract it to a new directory "foxitpdfsdk_10_1_win_c", which is shown in Figure 2-1.

NOTE: the highlighted rectangle in the figure is just the version of Foxit PDF SDK. Here the SDK version is 10.1, so it shows 10_1 in the package name.

The release package contains the following folders:

doc:	API references, developer guide
examples:	sample projects and demos
include:	header files for Foxit PDF SDK API
lib:	libraries and license files
res:	the default icc profile files used for output preview demo

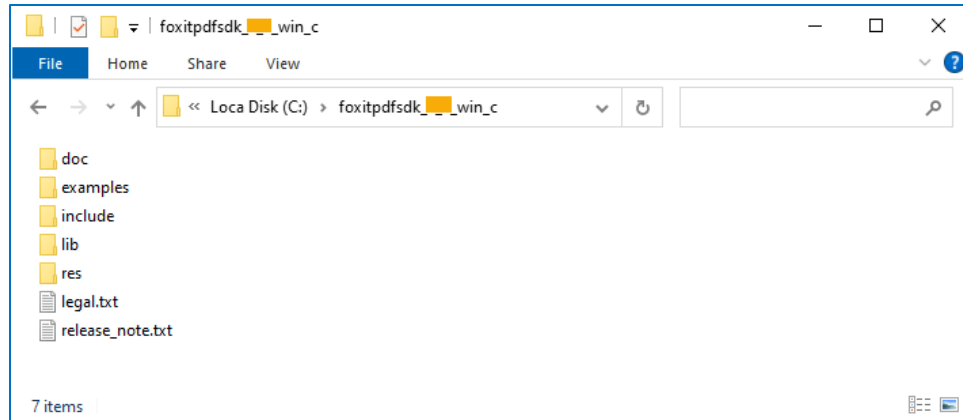


Figure 2-1

In the "examples" folder, there are two types of demos. "\examples\simple_demo" contains more than 30 demos that cover a wide range of PDF applications. "\examples\view_demo" contains a UI demo that realizes a lite PDF viewer.

2.3 How to run a demo

Foxit PDF SDK for C API provides several simple demos to show developers about how to effectively apply Foxit PDF SDK APIs to complete their applications.

To run a demo in Visual Studio (except security, signature, ocr, compliance, preflight, html2pdf, office2pdf, output preview, pdf2office, dwg2pdf and ofd demos which will be introduced later), you can follow the steps below:

- 1) Load the visual studio solution files "simple_demo_vs2010.sln" or "simple_demo_vs2015.sln" or "simple_demo_vs2017.sln" or "simple_demo_vs2019.sln" or "simple_demo_vs2022.sln" (depending on your Visual Studio version) in the "\examples\simple_demo" folder.
- 2) Build all the demos by clicking "Build > Build Solution". Alternatively, if you merely want to build a specific demo, you can right-click it and then choose "Build" or load the "*.vcxproj" file in the folder of a specific demo project and then build it.

After building, the executable file ".exe" will be generated in the "\examples\simple_demo\bin" folder. The names of the executable files depend on the build configurations.

- 3) Run a specific executable file by double-clicking it.

Some demos will generate output files (pdf, text or image files) to a folder named by the project name under "examples\simple_demo\output_files\" folder.

Note: If you want to see the detailed executing processes, you can run it in command line. Start "cmd.exe", navigate to "\examples\simple_demo\bin", and run a specific executable file.

Security demo

Before running **security** demo, you should install the certificates "foxit.cer" and "foxit_all.pfx" found in "\examples\simple_demo\input_files" folder.

- a) To install "foxit.cer", double-click it to start the certificate import wizard. Then select "Install certificate... > Next > Next > Finish".
- b) To install "foxit_all.pfx", double-click it to start the certificate import wizard. Then select "Next > Next > (Type the password "**123456**" for the private key in the textbox) and click Next > Next > Finish".
- c) Run the demo following the steps as the other demos.

Signature demo

Before running **signature** demo, you should ensure that the OpenSSL has been already installed in your machine. Download an OpenSSL source package from the OpenSSL website, or you can contact us directly, and then extract it and do the following:

- 1) Put the OpenSSL folder into the "include" folder which ensures that the OpenSSL header files included in the demos can be found.
- 2) Put the "libeay32.lib" library into the "lib" folder.
- 3) Run the demo following the steps as the other demos.

Note: We have verified that OpenSSL 1.1.1-stable version is available on the signature demo, you can replace it with the desired version, and maybe need to do some changes.

OCR and Compliance/Preflight demos

For **ocr** and **compliance/preflight** demos, you should build a resource directory at first, please contact Foxit support team or sales team to get the resource files packages. For more details about how to run the demos, please refer to section 3.34 "[OCR](#)" and section 3.35 "[Compliance](#)".

HTML to PDF demo

For html2pdf demo, you should contact Foxit support team or sales team to get the engine files package for converting from HTML to PDF at first. For more details about how to run the demo, please refer to section 3.37 "[HTML to PDF Conversion](#)".

Office to PDF demo

For office2pdf demo, you should make sure that Microsoft Office 2007 version or higher is already installed and the default Microsoft virtual printer is already set on your Windows system. Then, run the demo following the steps as the other demos.

Output Preview demo

For output preview demo, you should set the folder path which contains default icc profile files. For more details about how to run the demo, please refer to section 3.39 "[Output Preview](#)".

PDF to Office demo

For **pdf2office** demo, you should contact Foxit support team or sales team to get the engine files package for converting from PDF to office at first. For more details about how to run the demo, please refer to section 3.44 "[PDF to Office Conversion](#)".

DWG to PDF demo

For dwg2pdf demo, you should contact Foxit support team or sales team to get the engine files package for converting from DWG to PDF at first. For more details about how to run the demo, please refer to section 3.45 "[DWG to PDF Conversion](#)".

OFD demo

For ofd demo, you should contact Foxit support team or sales team to get the OFD engine files package at first. For more details about how to run the demo, please refer to section 3.46 "[OFD](#)".

2.4 How to create a simple project

In this section, we will show you how to use Foxit PDF SDK for Windows to create a simple project that renders the first page of a PDF to a bitmap and saves it as a JPG image. Please follow the steps below:

- 1) Open Visual Studio and create a new Win32 Console Application named "test_win".
- 2) Copy the "include" and "lib" folders from the "foxitpdfsdk_10_1_win_c" folder to the project "test_win" folder.
- 3) Add the "include" folder to your "Additional Include Directories". Right-click the *test_win* project in Solution Explorer, choose "Properties", and find "Configuration Properties > C/C++ > General > Additional Include Directories".
- 4) Add include header statements to the beginning of test_win.c.

```
#include "../include/fs_basictypes_c.h"
#include "../include/fs_common_c.h"
#include "../include/fs_pdfdoc_c.h"
#include "../include/fs_pdfpage_c.h"
#include "../include/fs_render_c.h"
```

- 5) Include Foxit PDF SDK library.

```
#if defined (_WIN64) // windows 64bit platforms.
    #if defined (_DEBUG)
        #pragma comment (lib, "../lib/fsdk_c_win64.lib")
    #else
        #pragma comment (lib, "../lib/fsdk_c_win64.lib")
    #endif

#elif defined (_WIN32) // windows 32bit platforms.
    #if defined (_DEBUG)
        #pragma comment (lib, "../lib/fsdk_c_win32.lib")
    #else
        #pragma comment (lib, "../lib/fsdk_c_win32.lib")
    #endif
#endif
```

- 6) Initialize the Foxit PDF SDK library. It is necessary for apps to initialize Foxit PDF SDK using a license before calling any APIs. The trial license files can be found in the "lib" folder.

```
const char* sn = "";
const char* key = "";
FSErrorCode code = FSDK_Library_Initialize(sn, key);
if (code != e_FSErrSuccess) {
    return false;
}
```

Note The value of "sn" can be got from "**gsdk_sn.txt**" (the string after "SN=") and the value of "key" can be got from "**gsdk_key.txt**" (the string after "Sign=").

- 7) Load a PDF document, and parse the first page of the document. Let us assume that you have already put a "Sample.pdf" to the "test_win\test_win" folder.

```
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0("Sample.pdf", &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) return 0;
```

```
FS_PDFPAGE_HANDLE page;  
FSDK_PDFDoc_GetPage(doc, 0, &page)  
FS_PROGRESSIVE_HANDLE progressivehandle;  
FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressivehandle);
```

- 8) Render a Page to a bitmap and save it as a JPG file.

```
float floatwidth, floatheight;  
FSDK_PDFPage_GetWidth(page, &floatwidth);  
FSDK_PDFPage_GetHeight(page, &floatheight);  
int width = (int)floatwidth;  
int height = (int)floatheight;  
FSRotation rotate;  
FSDK_PDFPage_GetRotation(page, &rotate);  
FSMatrix matrix;  
matrix.a = 1;  
matrix.b = 0;  
matrix.c = 0;  
matrix.d = 1;  
matrix.e = 0;  
matrix.f = 0;  
FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, rotate, &matrix);  
// Prepare a bitmap for rendering.  
FS_BITMAP_HANDLE bitmap;  
FSDK_Bitmap_Create(width, height, e_FSDIBArgb, NULL, 0, &bitmap);  
FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);  
// Render page.  
FS_RENDERER_HANDLE render;  
FSDK_Renderer_Create(bitmap, false, &render);  
FSDK_Renderer_StartRender(render, page, matrix, NULL, &progressivehandle);  
  
// Add the bitmap to image and save the image.  
FS_IMAGE_HANDLE image;  
FSDK_Image_Create(&image);  
FS_BOOL result = false;  
FSDK_Image_AddFrame(image, bitmap, &result);  
FSDK_Image_SaveAs(image, "testpage.jpg", &result);
```

- 9) Click "Build > Build Solution" to build the project. The executable file "test_win.exe" will be generated in "test_win\Debug" or "test_win\Release" folder depending on the build configurations.

10) Copy "fsdk_c_win32.dll" or "fsdk_c_win64.dll" in the "lib" folder to the output directory ("test_win\Debug" or "test_win\Release"). Please make sure that the "fsdk_c_win**.dll" architecture needs to match the platform target (Win32 or Win64) of the application.

11) Run the project. Choose one of the following:

- i. Click "Debug > Start Without Debugging" in Visual Studio to run the project, and the "testpage.jpg" will be generated in the "test_win\test_win" folder (same with "test_win.c").
- ii. Double-click the executable file "test_win.exe" to run the project. In this way, you should put the "Sample.pdf" to the same folder with the "test_win.exe", and the "testpage.jpg" will also be generated in the same folder.

The final contents of "test_win.c" is as follow:

```
#include "stdafx.h"

#include <iostream>
#include "../include/fs_basictypes_c.h"
#include "../include/fs_common_c.h"
#include "../include/fs_pdfdoc_c.h"
#include "../include/fs_pdfpage_c.h"
#include "../include/fs_render_c.h"

// Include Foxit PDF SDK library.
#ifdef _WIN64 // windows 64bit platforms.
    #ifdef _DEBUG
        #pragma comment(lib, "../lib/fsdk_c_win64.lib")
    #else
        #pragma comment(lib, "../lib/fsdk_c_win64.lib")
    #endif
#elif defined(_WIN32) // windows 32bit platforms.
    #ifdef _DEBUG
        #pragma comment(lib, "../lib/fsdk_c_win32.lib")
    #else
        #pragma comment(lib, "../lib/fsdk_c_win32.lib")
    #endif
#endif
```

```
int _tmain(int argc, _TCHAR* argv[])

{
    // The value of "sn" can be got from "gsdk_sn.txt" (the string after "SN=").
    // The value of "key" can be got from "gsdk_key.txt" (the string after "Sign=").
    const char* sn = " ";
    const char* key = " ";
    FSErrorCode code = FSDK_Library_Initialize(sn, key);
    if (code != e_FSErrSuccess) {
        return false;
    }

    // Load a PDF document, and parse the first page of the document.
    FS_PDFDOC_HANDLE doc;
    FSDK_PDFDoc_Create0("Sample.pdf", &doc);
    FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
    if (error_code != e_FSErrSuccess) return 0;
    FS_PDFPAGE_HANDLE page;
    FSDK_PDFDoc_GetPage(doc, 0, &page)
    FS_PROGRESSIVE_HANDLE progressivehandle;
    FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressivehandle);

    float floatwidth, floatheight;
    FSDK_PDFPage_GetWidth(page, &floatwidth);
    FSDK_PDFPage_GetHeight(page, &floatheight);
    int width = (int)floatwidth;
    int height = (int)floatheight;
    FSRotation rotate;
    FSDK_PDFPage_GetRotation(page, &rotate);
    FSMatrix matrix;
    matrix.a = 1;
    matrix.b = 0;
    matrix.c = 0;
    matrix.d = 1;
    matrix.e = 0;
    matrix.f = 0;
    FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, rotate, &matrix);

    // Prepare a bitmap for rendering.
```

```
FS_BITMAP_HANDLE bitmap;
FSDK_Bitmap_Create(width, height, e_FSDIBArgb, NULL, 0, &bitmap);
FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);
// Render page.
FS_RENDERER_HANDLE render;
FSDK_Renderer_Create(bitmap, false, &render);
FSDK_Renderer_StartRender(render, page, matrix, NULL, &progressivehandle);
// Add the bitmap to image and save the image.
FS_IMAGE_HANDLE image;
FSDK_Image_Create(&image);
FS_BOOL result = false;
FSDK_Image_AddFrame(image, bitmap, &result);
FSDK_Image_SaveAs(image, "testpage.jpg", &result);
return 0;
}
```

3. WORKING WITH SDK API

In this section, we will introduce a set of major features and list some examples for each feature to show you how to integrate powerful PDF capabilities with your applications using Foxit PDF SDK C API. You can refer to the API reference ^[2] to get more details about the APIs used in all of the examples.

3.1 Initialize Library

It is necessary for applications to initialize Foxit PDF SDK before calling any APIs. The function [FSDK_Library_Initialize](#) is provided to initialize Foxit PDF SDK. A license should be purchased for the application and pass unlock key and code to get proper supports. When there is no need to use Foxit PDF SDK any more, please call function [FSDK_Library_Release](#) to release it.

Note The parameter "sn" can be found in the "**gsdk_sn.txt**" (the string after "SN=") and the "key" can be found in the "**gsdk_key.txt**" (the string after "Sign=").

Example:

3.1.1 How to initialize Foxit PDF SDK

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
...

const char* sn = " ";
const char* key = " ";
FSErrorCode code = FSDK_Library_Initialize(sn, key);
if (code != e_FSErrSuccess) {
    return false;
}
```

3.2 Document

PDF document is represented by **FS_PDFDOC_HANDLE** object, which can be created with an existing PDF file from file path, memory buffer, a custom implemented ReaderCallback object and an input file stream. Then call function [FSDK_PDFDoc_Load](#) or [FSDK_PDFDoc_StartLoad](#) to load document content. It is used for document level operation, such as opening and closing files, getting page, metadata and etc.

Example:

3.2.1 How to create a PDF document from scratch

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create(&doc);
```

Note: It creates a new PDF document without any pages.

3.2.2 How to load an existing PDF document from file path

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0("Sample.pdf", &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code!= e_FSErrSuccess) return 0;
```

3.2.3 How to load an existing PDF document from a memory buffer

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
...

FILE* pFile = fopen(TEST_DOC_PATH"blank.pdf", "rb");
ASSERT_EQ(TRUE, NULL != pFile);
fseek(pFile, 0, SEEK_END);
long lFileSize = ftell(pFile);
char* buffer = malloc(lFileSize * sizeof(char));
memset(buffer, 0, sizeof(char)*lFileSize);
fseek(pFile, 0, SEEK_SET);
fread(buffer, sizeof(char), lFileSize, pFile);
fclose(pFile);
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create1(buffer, lFileSize, &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
Free(buffer);
if (error_code!= e_FSErrSuccess) return 0;
```

3.2.4 How to load an existing PDF document from a file read callback object

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
...

string wstring2string(const wchar_t *source, size_t source_size, char *dest, size_t dest_size) {
    char* curLocale = setlocale(LC_ALL, NULL);
    setlocale(LC_ALL, "chs");
```

```
memset(_Dest, 0, _Dsize);
wcstombs(dest, source, dest_size);
setlocale(LC_ALL, "C");
return dest;
}

FILE* file = NULL;
FS_BOOL is_large_file = FALSE;

FS_INT64 gGetSize(void* user_data) {
    if (!user_data) return 0;
    if (is_large_file) {
        long long size_long;
        _fseeki64(file, 0, SEEK_END);
        size_long = _ftelli64(file);
        return size_long;
    } else {
        fseek(file, 0, SEEK_END);
        return (FS_INT64)ftell(file);
    }
    return 0;
}

FS_BOOL gReadBlock(void* user_data, void* buffer, FS_INT64 offset, size_t size) {
    if (is_large_file) {
        long long read_size;
        _fseeki64(file, offset, SEEK_SET);

        read_size = fread(buffer, 1, size, file);
        return read_size == size ? TRUE : FALSE;
    } else {
        if (!file)
            return FALSE;
        if (0 != fseek(file, (long)offset, 0))
            return FALSE;
        if (0 == fread(buffer, size, 1, file))
            return FALSE;
        return TRUE;
    }
    return FALSE;
}

static void gRelease(void* user_data) {
    if (file) {
        fclose(file);
        file = NULL;
    }
    if (user_data)
        free(user_data);
}

...
```

```
const char* input_file = "Sample.pdf";
filereadercallback = (FSReaderCallback*)malloc(sizeof(FSReaderCallback));
filereadercallback->user_data = filereadercallback;
filereadercallback->GetSize = gGetSize;
filereadercallback->ReadBlock = gReadBlock;
filereadercallback->Release = gRelease;
_wfopen_s(&file, input_file, L"rb");
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create3(pFileRead, &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code!= e_FSErrSuccess) return 0;
```

3.2.5 How to load PDF document and get the first page of the PDF document

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0("Sample.pdf", &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code!= e_FSErrSuccess) return 0;
FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
FS_PROGRESSIVE_HANDLE progressive;
FSDK_PDFPage_StartParse(page, e_FSParsePageNormal, NULL, false, &progressive);
```

3.2.6 How to save a PDF to a file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0("Sample.pdf", &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code!= e_FSErrSuccess) return 0;
FSDK_PDFDoc_SaveAs(doc, "new_Sample.pdf", e_FSSaveFlagNoOriginal);
```

3.2.7 How to save a document into memory buffer by WriterCallback

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

// FileWriter for saving file to memory buffer.
...

CFX_BinaryBuf *binary_buffer_;
```

```

FS_INT64 gGetSizeFileWriter(void* user_data) {
    return binary_buffer_->GetSize();
}

FS_BOOL gFlushFileWriter(void* user_data) {
    return TRUE;
}

FS_BOOL gWriteBlockFileWriter(void* user_data, const void* buffer, FS_INT64 offset, size_t size) {
    return binary_buffer_->InsertBlock(offset,buffer,size);}

static void gReleaseFileWriter(void* user_data) {
}

// Assuming FS_PDFDOC_HANDLE doc has been loaded.
...
FSWriterCallback* filewritercallback = (FSWriterCallback*)malloc(sizeof(FSWriterCallback));
filewritercallback->user_data = filewritercallback;
filewritercallback->Flush = gFlushFileWriter;
filewritercallback->GetSize = gGetSizeFileWriter;
filewritercallback->WriteBlock = gWriteBlockFileWriter;
filewritercallback->Release = gReleaseFileWriter;

FS_PROGRESSIVE_HANDLE return_StartSaveAs;
FSDK_PDFDoc_StartSaveAs0(doc, filewriter, e_FSSaveFlagNoOriginal, NULL , &return_StartSaveAs);
...

```

3.3 Page

PDF page is represented by **FS_PDFPAGE_HANDLE** object. Page level APIs provide functions to parse, render, edit (includes creating, deleting, flattening and etc.) a page, retrieve PDF annotations, read and set the properties of a page, and etc. **FS_PDFPAGE_HANDLE** object is created by [FSDK_PDFDoc_GetPage](#) and needs to be cleared by [FSDK_PDFPage_Release](#). A PDF page needs to be parsed before it is rendered or processed.

Example:

3.3.1 How to get page size

```

#include "include/fs_basictypes_c.h"
#include "include/fs_pdfpage_c.h"
...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

float fWidth;
float fHeight;
FSDK_PDFPage_GetWidth(page, &fWidth);

```



```
FSDK_PDFPage_GetHeight(page, &fHeight);
int width= (int)fWidth;
int height= (int)fWidth;
```

3.3.2 How to calculate bounding box of page contents

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfpage_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

FSRectF ret;
ret.left = 0;
ret.bottom = 0;
ret.top = 0;
ret.right = 0;
FSDK_PDFPage_CalcContentBBox(page, e_FSCalcContentsBox, &ret);
...
```

3.3.3 How to create a PDF page and set the size

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_InsertPage(doc, index, PageWidth, PageHeight, &page);
```

3.3.4 How to delete a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

// Remove a PDF page by page index.
FS_BOOL return_value;
FSDK_PDFDoc_RemovePage(doc, index, &return_value);

// Remove a specified PDF page.
FSDK_PDFDoc_RemovePage0(doc, page, &return_value);
...
```

3.3.5 How to flatten a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfpage_c.h"
...
```

```
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

// Flatten all contents of a PDF page.
FS_BOOL return_Flatten;
FSDK_PDFPage_Flatten(page, true, e_FSFlattenAll, &return_Flatten);

// Flatten a PDF page without annotations.
FSDK_PDFPage_Flatten(page, true, e_FSFlattenNoAnnot, &return_Flatten);

// Flatten a PDF page without form controls.
FSDK_PDFPage_Flatten(page, true, e_FSFlattenNoFormControl, &return_Flatten);

// Flatten a PDF page without annotations and form controls (Equals to nothing to be flattened).
FSDK_PDFPage_Flatten(page, true, e_FSFlattenNoAnnot | e_FSFlattenNoFormControl, &return_Flatten);
...
```

3.3.6 How to get and set page thumbnails in a PDF document

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfpage_c.h"
...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

FS_BITMAP_HANDLE bmp;
FSDK_Bitmap_Create0(&bmp);
// Write bitmap data to the bmp object.
...
// Set thumbnails to the page.
FSDK_PDFPage_SetThumbnail(page, bmp);
// Load thumbnails in the page.
FS_BITMAP_HANDLE bitmap;
FSDK_PDFPage_LoadThumbnail(page, &bitmap);
...
```

3.4 Render

PDF rendering is realized through the Foxit renderer, a graphic engine that is used to render page to a bitmap or platform graphics device. Foxit PDF SDK provides APIs to set rendering options/flags, for example set flag to decide whether to render form fields and signature, whether to draw image anti-aliasing and path anti-aliasing. To do rendering, you can use the following APIs:

- To render page and annotations, first use function [FSDK_Renderer_SetRenderContentFlags](#) to decide whether to render page and annotation both or not, and then use function [FSDK_Renderer_StartRender](#) to do the rendering. Function [FSDK_Renderer_StartQuickRender](#) can also be used to render page but only for thumbnail purpose.
- To render a single annotation, use function [FSDK_Renderer_RenderAnnot](#).

- To render on a bitmap, use function [FSDK_Renderer_StartRenderBitmap](#).
- To render a reflowed page, use function [FSDK_Renderer_StartRenderReflowPage](#).

Widget annotation is always associated with form field and form control in Foxit PDF SDK. For how to render widget annotations, here is a recommended flow:

- After loading a PDF page, first render the page and all annotations in this page (including widget annotations).
- Then, if use [FS_FILLER_HANDLE](#) object to fill the form, the function [FSDK_Filler_Render](#) should be used to render the focused form control instead of the function [FSDK_Renderer_RenderAnnot](#).

Example:

3.4.1 How to render a page to a bitmap

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_render_c.h"

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

float fWidth;
float fHeight;
FSDK_PDFPage_GetWidth(page, &fWidth);
FSDK_PDFPage_GetHeight(page, &fHeight);
int width = (int)fWidth;
int height = (int)fHeight;

FSMatrix matrix;
FSRotation rotation;
FSDK_PDFPage_GetRotation(page, &rotation);
FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, rotation, &matrix);

// Prepare a bitmap for rendering.
FS_BITMAP_HANDLE bitmap;
FSDK_Bitmap_Create(width, height, e_FSDIBArgb, NULL, 0, &bitmap);
FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);
// Render page.
FS_RENDERER_HANDLE renderer;
FSDK_Renderer_Create(bitmap, false, &renderer);
FS_PROGRESSIVE_HANDLE progessive;
FSDK_Renderer_StartRender(renderer, page, matrix, NULL, &progessive);
...
```

3.4.2 How to render page and annotation

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_render_c.h"

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
float fWidth;
float fHeight;
FSDK_PDFPage_GetWidth(page, &fWidth);
FSDK_PDFPage_GetHeight(page, &fHeight);
int width = (int)fWidth;
int height = (int)fHeight;
FSMatrix matrix;
FSRotation rotation;
FSDK_PDFPage_GetRotation(page, &rotation);
FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, rotation, &matrix);

// Prepare a bitmap for rendering.
FS_BITMAP_HANDLE bitmap;
FSDK_Bitmap_Create(width, height, e_FSDIBArgb, NULL, 0, &bitmap);
FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);
FS_RENDERER_HANDLE renderer;
FSDK_Renderer_Create(bitmap, false, &renderer);
FS_UINT32 dwRenderFlag = e_FSRenderAnnot | e_FSRenderPage;
FSDK_Renderer_SetRenderContentFlags(render, dwRenderFlag);
FS_PROGRESSIVE_HANDLE progreessive;
FSDK_Renderer_StartRender(render, page, matrix, NULL, &progreessive);
...
```

3.5 Attachment

In Foxit PDF SDK, attachments are only referred to attachments of documents rather than file attachment annotation, which allow whole files to be encapsulated in a document, much like email attachments. PDF SDK provides applications APIs to access attachments such as loading attachments, getting attachments, inserting/removing attachments, and accessing properties of attachments.

Example:

3.5.1 How to export the embedded attachment file from a PDF and save it as a single file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfattachments_c.h"
```

```
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

// Get information of attachments.
FS_ATTACHMENTS_HANDLE attachments;
FS_PDFNAMETREE_HANDLE pdfnametree;
FSDK_PDFNameTree_Create0(&pdfnametree);
FSDK_Attachments_Create(doc, pdfnametree, &attachments);
int count;
FSDK_Attachments_GetCount(attachments, &count);
for (int i = 0; i < count; i++) {
    FS_WSTR key;
    FSDK_WStr_Init(key);
    FSDK_Attachments_GetKey(attachments, i, &key);
    FS_FILESPEC_HANDLE file_spec;
    FSDK_Attachments_GetEmbeddedFile(attachments, (const wchar_t*)key.str, &file_spec);
    FSDK_WStr_Clear(key);
    FS_BOOL return_value;
    FSDK_FileSpec_IsEmpty(file_spec, &return_value);
    if (!return_value) {
        FS_WSTR name;
        FSDK_WStr_Init(name);
        FS_BOOL isEmbedded;
        FSDK_FileSpec_GetFileName(file_spec, &name);
        FSDK_FileSpec_IsEmbedded(file_spec, &isEmbedded);
        if (isEmbedded) {
            const wchar_t *exFilePath = "output_directory";
            FS_BOOL bExportStatus;
            FSDK_FileSpec_ExportToFile(file_spec, exFilePath, &bExportStatus);
        }
        FSDK_WStr_Clear(name);
    }
}
...

```

3.5.2 How to remove all the attachments of a PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfattachments_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

// Get information of attachments.
FS_ATTACHMENTS_HANDLE attachments;
FS_PDFNAMETREE_HANDLE pdfnametree;
FSDK_PDFNameTree_Create0(&pdfnametree);
FSDK_Attachments_Create(doc, pdfnametree, &attachments);
int count;
FSDK_Attachments_GetCount(attachments, &count);
for (int i = 0; i < count; i++) {

```

```

FS_WSTR key;
FSDK_WStr_Init(key);
FSDK_Attachments_GetKey(attachments, i, &key);
FS_BOOL isRemove;
FSDK_Attachments_RemoveEmbeddedFile( attachments, (const wchar_t*)key.str, &isRemove);
FSDK_WStr_Clear(key);
}
...

```

3.6 Text Page

Foxit PDF SDK provides APIs to extract, select, search and retrieve text in PDF documents. PDF text contents are stored in [FS_TEXTPAGE_HANDLE](#) objects which are related to a specific page.

[FS_TEXTPAGE_HANDLE](#) can be used to retrieve information about text in a PDF page, such as single character, single word, text content within specified character range or rectangle and so on. It also can be used to construct objects of other text related classes to do more operations for text contents or access specified information from text contents:

- To search text in text contents of a PDF page, construct a [FS_TEXTSEARCH_HANDLE](#) object with [FS_TEXTPAGE_HANDLE](#) object.
- To access text such like hypertext link, construct a [FS_PAGETEXTLINKS_HANDLE](#) object with [FS_TEXTPAGE_HANDLE](#) object.

Example:

3.6.1 How to extract text from a PDF page

```

#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_search_c.h"

...

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

// Get the text page object.
FS_TEXTPAGE_HANDLE text_page;
FSDK_TextPage_Create(page, e_FSTextParseFlagsParseTextNormal, &text_page);
int count;
FSDK_TextPage_GetCharCount(text_page, &count);
if (count > 0) {
    FS_WSTR text;
    FSDK_TextPage_GetChars(text_page, 0, -1, &text);
    FS_BSTR text_bstr;
    FSErrorCode code = FSDK_WStr_UTF8Encode(text, &text_bstr);
    fwrite(text_bstr.str, sizeof(char), text_bstr.len, file);
}

```

```
FSDK_WStr_Clear(text);  
}  
...
```

3.6.2 M

3.6.3 How to get the text within a rectangle area in a PDF

```
#include "include/fs_basictypes_c.h"  
#include "include/fs_common_c.h"  
#include "include/fs_pdfdoc_c.h"  
#include "include/fs_search_c.h"  
...  
  
FSRectF rect;  
rect.left = 90;  
rect.right = 450;  
rect.top = 595;  
rect.bottom = 580;  
FS_TEXTPAGE_HANDLE text_page;  
FSDK_TextPage_Create(page, e_FSTextParseFlagsParseTextNormal, &text_page);  
FS_WSTR text;  
FSDK_TextPage_GetTextInRect(textPage, rect, &text);  
...
```

3.7 Text Search

Foxit PDF SDK provides APIs to search text in a PDF document, a XFA document, a text page or in a PDF annotation's appearance. It offers functions to do a text search and get the searching result:

- To specify the searching pattern and options, use functions [FSDK_TextSearch_SetPattern](#), [FSDK_TextSearch_SetStartPage](#) (only useful for a text search in PDF document), [FSDK_TextSearch_SetEndPage](#) (only useful for a text search in PDF document) and [FSDK_TextSearch_SetSearchFlags](#).
- To do the searching, use function [FSDK_TextSearch_FindNext](#) or [FSDK_TextSearch_FindPrev](#).
- To get the searching result, use function [FSDK_TextSearch_GetMatchXXX\(\)](#).

Example:

3.7.1 How to search a text pattern in a PDF

```
#include "include/fs_basictypes_c.h"  
#include "include/fs_common_c.h"  
#include "include/fs_pdfdoc_c.h"  
#include "include/fs_pdfpage_c.h"  
#include "include/fs_search_c.h"  
...
```

```
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

// Search for all pages of doc.
FS_TEXTSEARCH_HANDLE search;
FSDK_TextSearch_Create(doc, NULL, e_FSTextParseFlagsParseTextNormal, &search);

int start_index = 0;
int end_index = 0;
int index = 0;
FSDK_PDFDoc_GetPageCount(doc, &index);
end_index = index - 1;
FS_BOOL return_value1 = false;
FSDK_TextSearch_SetStartPage(search, start_index, &return_value1);
FS_BOOL return_value2 = false;
FSDK_TextSearch_SetEndPage(search, end_index, &return_value2);

const wchar_t* pattern = L"Foxit";
FS_BOOL return_value3;
FSDK_TextSearch_SetPattern(search, pattern, &return_value3);

FS_UINT32 flags = e_FSSearchFlagsSearchNormal;
FS_BOOL return_value4;
FSDK_TextSearch_SetSearchFlags(search, flags, &return_value4);
...

int match_count = 0;
FS_BOOL return_value5;
FSDK_TextSearch_FindNext(search, &return_value5);
while (return_value5) {
    FSRectF *rect_array = NULL;
    FS_UINT32 return_array_length;
    FSDK_TextSearch_GetMatchRects(search, rect_array, &return_array_length);
    match_count++;
    rect_array = (FSRectF*)malloc(length * sizeof(FSRectF));
    FSDK_TextSearch_GetMatchRects(search, rect_array, &return_array_length);
    FSDK_TextSearch_FindNext(search, &return_value5);
    free(rect_array);
}
...
```

3.8 Search and Replace

The Search and Replace feature allows you to search for specific text content within a PDF document and replace it with new content.

3.8.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'AdvEdit' module permission in the license key

SDK Version: Foxit PDF SDK (C, C++, C#, Java, Python, Objective-C) 9.0 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.8.2 How to work with the search and replace function

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_searchreplace_c.h"

FSErrorCode error_code = e_FSErrSuccess;
FS_TEXTSEARCHREPLACE_HANDLE searchreplace = NULL;
FSFindOption find_option;
FSReplaceCallback* replacecallback = NULL;
error_code = FSDK_PDFDoc_Load(doc, NULL);

// Instantiate a TextSearchReplace object.
FSDK_TextSearchReplace_Create(doc, &searchreplace);

// Set replacing callback function.
FSDK_TextSearchReplace_SetReplaceCallback(searchreplace, &replacecallback);

// Set keywords and page index to do searching and replacing.
FSDK_TextSearchReplace_SetPattern(searchreplace, L"PDF", 0, find_option);

// Replace with new text.
while (result) FSDK_TextSearchReplace_ReplaceNext(searchreplace, L"PDC", &result);
```

3.9 Text Link

In a PDF page, some text contents that represent a hypertext link to a website or a resource on the internet, or an email address are the same with common texts. Prior to text link processing, user should first call `FSDK_PageTextLinks_GetTextLink` to get a textlink object.

Example:

3.9.1 How to retrieve hyperlinks in a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_search_c.h"
...

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

// Get the text page object.
```

```
FS_TEXTPAGE_HANDLE text_page;
FSDK_TextPage_Create(page, e_FSTextParseFlagsParseTextNormal, &text_page);
FS_PAGETEXTLINKS_HANDLE pageTextLink;
FSDK_PageTextLinks_Create(text_page, &pageTextLink);
FS_TEXTLINK_HANDLE textLink;
FSDK_PageTextLinks_GetTextLink(pageTextLink, index, &textLink);
FS_WSTR strURL;
FSDK_TextLink_GetURI(textLink, &strURL);
...
```

3.10 Bookmark

Foxit PDF SDK provides navigational tools called Bookmarks to allow users to quickly locate and link their point of interest within a PDF document. PDF bookmark is also called outline, and each bookmark contains a destination or actions to describe where it links to. It is a tree-structured hierarchy, so function [FSDK_PDFDoc_GetRootBookmark](#) must be called first to get the root of the whole bookmark tree before accessing to the bookmark tree. Here, "root bookmark" is an abstract object which can only have some child bookmarks without next sibling bookmarks and any data (includes bookmark data, destination data and action data). It cannot be shown on the application UI since it has no data. Therefore, a root bookmark can only call function [FSDK_Bookmark_GetFirstChild](#).

After the root bookmark is retrieved, following functions can be called to access other bookmarks:

- To access the parent bookmark, use function [FSDK_Bookmark_GetParent](#).
- To access the first child bookmark, use function [FSDK_Bookmark_GetFirstChild](#).
- To access the next sibling bookmark, use function [FSDK_Bookmark_GetNextSibling](#).
- To insert a new bookmark, use function [FSDK_Bookmark_Insert](#).
- To move a bookmark, use function [FSDK_Bookmark_MoveTo](#).

Example:

3.10.1 How to find and list all bookmarks of a PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_filespec_c.h"
#include "include/fs_bookmark_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_BOOKMARK_HANDLE root;
FSDK_PDFDoc_GetRootBookmark(doc, &root);
FS_BOOKMARK_HANDLE first_bookmark;
```

```
FSDK_Bookmark_GetFirstChild(root, &first_bookmark);

if (first_bookmark != null)
{
    TraverseBookmark(first_bookmark, 0);
}

Private void TraverseBookmark(FS_BOOKMARK_HANDLE root, int iLevel)
{
    if (root != null)
    {
        FS_BOOKMARK_HANDLE child;
        FSDK_Bookmark_GetFirstChild(root, &child);
        while (child != null)
        {
            TraverseBookmark(child, iLevel + 1);
            FSDK_Bookmark_GetNextSibling(child, &child);
        }
    }
}
...
```

3.10.2 How to insert a new bookmark

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_filespec_c.h"
#include "include/fs_bookmark_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"

int string2wstring(const char *source, size_t source_size, wchar_t *dest, size_t dest_size) {
    const char *_source;
    wchar_t *_dest;
    if (!source || !dest) return 0;
    _source = source;
    _dest = dest;
    setlocale(LC_ALL, "chs");
    mbstowcs(_dest, _source, _Dsize);
    setlocale(LC_ALL, "C");
    return (int)dest_size;
}

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_BOOKMARK_HANDLE root;
FSDK_PDFDoc_GetRootBookmark(doc, &root);
FS_BOOL return_value;
FSDK_Bookmark_IsEmpty(root, &return_value);
if (return_value)
{
```

```
FSDK_PDFDoc_CreateRootBookmark(doc, &root);
}
FS_DESTINATION_HANDLE dest;
FSDK_Destination_CreateFitPage(doc, 0, &dest);
char str[128];
sprintf(str, "A bookmark to a page (index: %d)", i);
wchar_t ws_title[MAX_FILE_PATH];
FSErrorCode error_code;
string2wstring(path, MAX_FILE_PATH, ws_title, MAX_FILE_PATH);
FS_BOOKMARK_HANDLE child;
FSDK_Bookmark_Insert(root, ws_title, e_FSPosLastChild, &child);
FSDK_Bookmark_SetDestination(child, dest);
FSDK_Bookmark_SetColor(child, 0xF68C21);
```

3.10.3 How to create a table of contents based on bookmark information in PDFs

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_filespec_c.h"
#include "include/fs_bookmark_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"

void AddTOCToPDF(FS_PDFDOC_HANDLE& doc) {
    // Set the table of contents configuration.
    int depth = 0;
    FSDK_PDFDoc_GetBookmarkLevelDepth(doc, &depth);
    int* intarray = (int*)malloc(depth * sizeof(int));
    if (depth > 0) {
        for (int i = 1; i <= depth; i++) {
            intarray[i - 1] = i;
        }
    }
    FS_WSTR title;
    title.str = NULL;
    title.len = 0;
    FSTableOfContentsConfig toc_config;
    toc_config.bookmark_level_array = intarray;
    toc_config.bookmark_level_array_length = depth;
    toc_config.title = title;
    toc_config.is_show_serial_number = true;
    toc_config.include_toc_pages = false;

    // Add the table of contents.
    FSDK_PDFDoc_AddTableOfContents0(doc, toc_config);
    free(intarray);
}
```

3.11 Form (AcroForm)

PDF currently supports two different forms for gathering information interactively from the user - AcroForms and XFA forms. Acroforms are the original PDF-based fillable forms, based on the PDF architecture. Foxit PDF SDK provides APIs to view and edit form field programmatically. Form fields are commonly used in PDF documents to gather data. Foxit PDF SDK offers functions to retrieve form fields or form controls, import/export form data and other features, for example:

- To retrieve form fields, please use functions [FSDK_Form_GetFieldCount](#) and [FSDK_Form_GetField](#).
- To retrieve form controls from a PDF page, please use functions [FSDK_Form_GetControlCount](#) and [FSDK_Form_GetControl](#).
- To import form data from an XML file, please use function [FSDK_Form_ImportFromXML](#); to export form data to an XML file, please use function [FSDK_Form_ExportToXML](#).
- To retrieve form filler object, please use function [FSDK_Form_GetFormFiller](#).

To import form data from a FDF/XFDF file or export such data to a FDF/XFDF file, please refer to functions [FSDK_PDFDoc_ImportFromFDF](#) and [FSDK_PDFDoc_ExportToFDF](#).

Example:

3.11.1 How to load the forms in a PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_BOOL hasForm;
FSDK_PDFDoc_HasForm(doc, &hasForm);
if(hasForm)
    FS_FORM_HANDLE form;
    FSDK_Form_Create(doc, &form);
...
```

3.11.2 How to count form fields and get/set the properties

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"
...
```

```
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_FORM_HANDLE form;
FSDK_Form_Create(doc, &form);
int countFields = 0;
FSDK_Form_GetFieldCount(form, NULL, &countFields);
for (int i = 0; i < nFieldCount; i++)
{
    FS_FIELD_HANDLE field;
    FSDK_Form_GetField(form, i, NULL, &field);
    FSFieldType type;
    FSDK_Field_GetType(field, &type);
    FS_WSTR org_alternateName;
    FSDK_Field_GetAlternateName(field, &org_alternateName);
    FSDK_Field_SetAlternateName(field, L"signature");
}
```

3.11.3 How to export the form data in a PDF to a XML file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
char* wstring2string(const wchar_t *source, size_t source_size, char *dest, size_t dest_size) {
    char* curLocale = setlocale(LC_ALL, NULL);
    setlocale(LC_ALL, "chs");
    memset(dest, 0, dest_size);
    wcstombs(dest, source, dest_size);
    setlocale(LC_ALL, "C");
    return dest;
}
...
FS_FORM_HANDLE form;
FSDK_Form_Create(doc, &form);
...
FS_BOOL return_value;
xmlFileSize = wcslen(XMLFilePath);
destXMLsize = xmlFileSize + 1;
char* destXML = (char*)malloc(dest_size);
wstring2string(XMLFilePath, xmlFileSize, destXML, destXMLsize);
FSDK_Form_ExportToXML(form, , &return_value);
free(destXML);
...
```

3.11.4 How to import form data from a XML file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"
```

```
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_FORM_HANDLE form;
FSDK_Form_Create(doc, &form);

...
FS_BOOL return_value;
wstring2string(XMLFilePath, xmlFileSize, destXML, destXMLsize);
FSDK_Form_ImportFromXML(form, destXML, &return_value);
...
```

3.11.5 How to get coordinates of a form field

1. Load PDF file by PDFDoc.
2. Traverse the form fields of the PDFDoc to get the field object of form.
3. Traverse the form controls of the field object to get the form control object.
4. Get the related widget annotation object by form control.
5. Call the GetRect of the widget annotation object to get the coordinate of the form.

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"

...
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
if (error_code != e_FSErrSuccess) return 1;

error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) {
    FSDK_PDFDoc_Release(doc);
    return 1;
}
FS_BOOL isHaveForm;
FSDK_PDFDoc_HasForm(doc, &isHaveForm);
if (!isHaveForm) return 1;
FS_FORM_HANDLE form;
FSDK_Form_Create(doc, &form);
int fieldCount = 0;
FSDK_Form_GetFieldCount(form, NULL, &fieldCount);
for (int i = 0; i < fieldCount; i++) {
    FS_FIELD_HANDLE field;
    FSDK_Form_GetField(form, i, NULL, &field);
    FS_BOOL isEmpty;
    FSDK_Field_IsEmpty(field, &isEmpty);
    if (isEmpty) continue;
    int controlCount = 0;
    FSDK_Field_GetControlCount(field, &controlCount);
    for (int j = 0; j < controlCount; j++) {
        FS_CONTROL_HANDLE control;
```

```
FSDK_Field_GetControl(field, j, &control);
FS_WIDGETANNOT_HANDLE widget;
FSDK_Control_GetWidget(control, &widget);
FSRectF rect;
//Get rectangle of the annot widget.
FSDK_Annot_GetRect(widget, &rect);
}
}
FSDK_Form_Release(form);
FSDK_PDFDoc_Release(doc);
...
```

3.12 XFA Form

XFA (XML Forms Architecture) forms are XML-based forms, wrapped inside a PDF. The XML Forms Architecture provides a template-based grammar and a set of processing rules that allow users to build interactive forms. At its simplest, a template-based grammar defines fields in which a user provides data.

Foxit PDF SDK provides APIs to render the XFA form, fill the form, export or import form's data.

Note:

- Foxit PDF SDK provides two structs *FSAppProviderCallback* and *FSDocProviderCallback* to represent the callback objects as an XFA document provider and an XFA application provider respectively.
- To use the XFA form feature, please make sure the license key has the permission of the 'XFA' module.

Example:

3.12.1 How to load XFADoc and represent an Interactive XFA form

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"

#include "include/fs_pdfform_c.h"
#include "include/fs_xfa_c.h"

FSAppProviderCallback* pXFAAppHandler = (FSAppProviderCallback*)malloc(sizeof(FSAppProviderCallback));
pXFAAppHandler->user_data = xfa_app_provider;
pXFAAppHandler->Release = gRelease;
pXFAAppHandler->GetAppInfo = gGetAppInfo;
pXFAAppHandler->Beep = gBeep;
pXFAAppHandler->MsgBox = gMsgBox;
```



```
pXFAAppHandler ->FSResponse = gResponse;
pXFAAppHandler ->DownloadUrl = gDownloadUrl;
pXFAAppHandler ->PostRequestURL = gPostRequestURL;
pXFAAppHandler ->PutRequestURL = gPutRequestURL;
pXFAAppHandler ->LoadString = gLoadString;
pXFAAppHandler ->ShowFileDialog = gShowFileDialog;
FSDK_Library_RegisterXFAAppProviderCallback(pXFAAppHandler);
swprintf_s(input_file, MAX_FILE_PATH, L"%s xfa_dynamic.pdf ", input_path);
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FErrSuccess) {
    return 1;
}

xfa_doc_handler = (FSDocProviderCallback*)malloc(sizeof(FSDocProviderCallback));
xfa_doc_handler->user_data = xfa_doc_handler;
xfa_doc_handler->Release = gXFADocProviderRelease;
xfa_doc_handler->InvalidateRect = gInvalidateRect;
xfa_doc_handler->DisplayCaret = gDisplayCaret;
xfa_doc_handler->GetPopupPos = gGetPopupPos;
xfa_doc_handler->PopupMenu = gPopupMenu;
xfa_doc_handler->GetCurrentPage = gGetCurrentPage;
xfa_doc_handler->SetCurrentPage = gSetCurrentPage;
xfa_doc_handler->SetChangeMark = gSetChangeMark;
xfa_doc_handler->GetTitle = gGetTitle;
xfa_doc_handler->ExportData = gExportData;
xfa_doc_handler->ImportData = gImportData;
xfa_doc_handler->GotoURL = gGotoURL;
xfa_doc_handler->Print = gPrint;
xfa_doc_handler->GetHighlightColor = gGetHighlightColor;
xfa_doc_handler->SubmitData = gSubmitData;
xfa_doc_handler->PageViewEvent = gPageViewEvent;
xfa_doc_handler->WidgetEvent = gWidgetEvent;
FS_XFADOC_HANDLE xfa_doc;
FSDK_XFADoc_Create(doc, xfa_doc_handler, &xfa_doc);
FS_PROGRESSIVE_HANDLE progressive;
FSDK_XFADoc_StartLoad(xfa_doc, NULL, &progressive);
...
```

3.12.2 How to export and import XFA form data

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfform_c.h"
#include "include/fs_xfa_c.h"

// Assuming FSXFADoc xfa_doc has been loaded.

FS_BOOL return_ExportData;
FSDK_XFADoc_ExportData(xfa_doc, L"xfa_form.xml", e_FSExportDataTypeXML, &return_ExportData);
```

```
FSDK_XFADoc_ResetForm(xfa_doc);
FS_BOOL return_SaveAs;
FSDK_PDFDoc_SaveAs(doc, L"xfa_dynamic_resetform.pdf", e_FSSaveFlagsSaveFlagNormal, &return_SaveAs);

FS_BOOL return_ImportData;
FSDK_XFADoc_ImportData(xfa_doc, L"xfa_form.xml", &return_ImportData);
FS_BOOL return_SaveAs2;
FSDK_PDFDoc_SaveAs(doc, L"xfa_dynamic_importdata.pdf", e_FSSaveFlagsSaveFlagNormal, &return_SaveAs2);
...
```

3.13 Form Design

Fillable PDF forms (AcroForm) are especially convenient for preparation of various applications, such as taxes and other government forms. Form design provides APIs to add or remove form fields (Acroform) to or from a PDF file. Designing a form from scratch allows developers to create the exact content and layout of the form they want.

Example:

3.13.1 How to add a text form field to a PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_pdfform_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

// Add text field
FS_CONTROL_HANDLE control;
FSRectF rect;
rect.left = 50;
rect.bottom = 600;
rect.right = 90;
rect.top = 640;
FSDK_Form_AddControl(form, page, L"Text Field0", e_FSTypeTextField, rect, &control);
FS_FIELD_HANDLE field;
FSDK_Control_GetField(control, &field);
FSDK_Field_SetValue(field, L"3");
// Update text field's appearance.
FS_WIDGETANNOT_HANDLE widget;
FSDK_Control_GetWidget(control, &widget);
FS_BOOL result;
FSDK_Annot_ResetAppearanceStream(widget, &result);

rect.left = 100;
rect.bottom = 600;
```

```

rect.right = 140;
rect.top = 640;
FS_CONTROL_HANDLE control1;
FSDK_Form_AddControl(form, page, L"Text Field1", e_FSTypeTextField, rect, &control1);
FS_FIELD_HANDLE field1;
FSDK_Control_GetField(control, &field1);
FSDK_Field_SetValue(field1, L"123");
// Update text field's appearance.
FS_WIDGETANNOT_HANDLE widget1;
FSDK_Control_GetWidget(control1, &widget1);
FS_BOOL result1;
FSDK_Annot_ResetAppearanceStream(widget1, &result1);
...

```

3.13.2 How to remove a text form field from a PDF

```

#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_pdfform_c.h"
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_FORM_HANDLE form;
FSDK_Form_Create(doc, form);
const wchar_t* filter = L"text1";
int countFields;
FSDK_Form_GetFieldCount(form, NULL, &countFields);
for (int i = 0; i < countFields; i++)
{
    FS_FIELD_HANDLE field;
    FSDK_Form_GetField(form, i, filter, &field);
    FSFieldType type;
    FSDK_Field_GetType(field, &type);
    if (type == e_FSTypeTextField) {
        FSDK_Form_RemoveField(form, field);
    }
}
...

```

3.14 Annotations

3.14.1 General

An annotation associates an object such as note, line, and highlight with a location on a page of a PDF document. It provides a way to interact with users by means of the mouse and keyboard. PDF includes a wide variety of standard annotation types as listed in Table 3-1. Among these annotation types, many of them are defined as markup annotations for they are used primarily to mark up PDF documents. These annotations have text that appears as part of the annotation and may be displayed

in other ways by a conforming reader, such as in a Comments pane. The 'Markup' column in Table 3-1 shows whether an annotation is a markup annotation.

Foxit PDF SDK supports most annotation types defined in PDF reference [1]. PDF SDK provides APIs of annotation creation, properties access and modification, appearance setting and drawing.

Table 3-1

Annotation type	Description	Markup	Supported by SDK
Text(Note)	Text annotation	Yes	Yes
Link	Link Annotation	No	Yes
FreeText (TypeWriter/TextBox/Callout)	Free text annotation	Yes	Yes
Line	Line annotation	Yes	Yes
Square	Square annotation	Yes	Yes
Circle	Circle annotation	Yes	Yes
Polygon	Polygon annotation	Yes	Yes
PolyLine	PolyLine annotation	Yes	Yes
Highlight	Highlight annotation	Yes	Yes
Underline	Underline annotation	Yes	Yes
Squiggly	Squiggly annotation	Yes	Yes
StrikeOut	StrikeOut annotation	Yes	Yes
Stamp	Stamp annotation	Yes	Yes
Caret	Caret annotation	Yes	Yes
Ink(pencil)	Ink annotation	Yes	Yes
Popup	Popup annotation	No	Yes
File Attachment	FileAttachment annotation	Yes	Yes
Sound	Sound annotation	Yes	No
Movie	Movie annotation	No	No
Widget*	Widget annotation	No	Yes
Screen	Screen annotation	No	Yes
PrinterMark	PrinterMark annotation	No	No
TrapNet	Trap network annotation	No	No
Watermark*	Watermark annotation	No	Yes
3D	3D annotation	No	No
Redact	Redact annotation	Yes	Yes

Note:

1. The annotation types of widget and watermark are special. They aren't supported in the module of 'Annotation'. The type of widget is only used in the module of 'form filler' and the type of watermark only in the module of 'watermark'.
2. Foxit SDK supports a customized annotation type called PSI (pressure sensitive ink) annotation that is not described in PDF reference [1]. Usually, PSI is for handwriting features and Foxit SDK treats it as PSI annotation so that it can be handled by other PDF products.

Example:

3.14.1.1 How to add a link annotation to a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
#include "include/fs_pdfpage_c.h"

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
// Assuming the annots in the page have been loaded.

// Add link annotation.
FSRectF rectf;
rectf.left = 350;
rectf.bottom = 350;
rectf.right = 380;
rectf.top = 400;
FS_ANNOT_HANDLE annot;
FSDK_PDFPage_AddAnnot(page, e_FSLink, rectf, &annot);
FS_LINKANNOT_HANDLE link;
FSDK_Link_Create0(annot, &link);
FSDK_Link_SetHighlightingMode(link, e_FSHighlightingToggle);
...
```

3.14.1.2 How to add a highlight annotation to a page and set the related annotation properties

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
#include "include/fs_pdfpage_c.h"

int string2wstring(const char *source, size_t source_size, wchar_t *dest, size_t dest_size) {
    const char *_source;
    wchar_t *_dest;
    if (!source || !dest) return 0;
    _source = source;
```

```
_dest = dest;
setlocale(LC_ALL, "chs");
mbstowcs(_dest, _source, _Dsize);
setlocale(LC_ALL, "C");
return (int)dest_size;
}
...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
// Assuming the annots in the page have been loaded.

// Add highlight annotation.
FSRectF rectf;
rectf.left = 10;
rectf.bottom = 450;
rectf.right = 100;
rectf.top = 550;
FS_ANNOT_HANDLE annot;
FSDK_PDFPage_AddAnnot(page, e_FSHighlight, rectf, &annot);
FS_HIGHLIGHTANNOT_HANDLE highlight;
FSDK_Highlight_Create0(annot, &highlight);
FSDK_Annot_SetContent(highlight, L"Highlight");
FSQuadPoints quad_points;
FSPointF point1;
point1.x = 10;
point1.y = 500;
quad_points.first = point1;
FSPointF point2;
point2.x = 90;
point2.y = 500;
quad_points.second = point2;
FSPointF point3;
point3.x = 10;
point3.y = 480;
quad_points.third = point3;
FSPointF point4;
point4.x = 90;
point4.y = 480;
quad_points.fourth = point4;
FSQuadPoints *quad_points_array = malloc(1 * sizeof(FSQuadPoints));
quad_points_array[0] = quad_points;
FSDK_TextMarkup_SetQuadPoints(highlight, quad_points_array, 1);
free(quad_points_array);
FSDK_Markup_SetSubject(highlight, L"Highlight");
FSDK_Markup_SetTitle(highlight, L"Foxit SDK");
FSDK_Markup_SetCreationDateTime(highlight, GetLocalDateTime());
FSDK_Annot_SetModifiedDateTime(highlight, GetLocalDateTime());
FSDK_Annot_SetUniqueID(highlight, WRandomUID());
// Appearance should be reset.
FS_BOOL return_code;
FSDK_Annot_ResetAppearanceStream(highlight, &return_code);
FSDK_Highlight_Release(highlight);
FSDK_Annot_Release(annot);
```

...

3.14.1.3 How to set the popup information when creating markup annotations

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
#include "include/fs_pdfpage_c.h"
```

```
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
// Assuming the annots in the page have been loaded.
```

```
// Create a new note annot and set the properties for it.
FSRectF rectf;
rectf.left = 10;
rectf.bottom = 350;
rectf.right = 50;
rectf.top = 400;
FS_ANNOT_HANDLE annot;
FSDK_PDFPage_AddAnnot(page, e_FSNote, rectf, &annot);
FS_NOTEANNOT_HANDLE note;
FSDK_Note_Create0(annot, &note);
FSDK_Note_SetIconName(note, "Comment");
FSDK_Markup_SetSubject(note, L"Note");
FSDK_Markup_SetTitle(note, L"Foxit SDK");
FSDK_Annot_SetContent(note, L"Note annotation.");
FSDK_Markup_SetCreationDateTime(note, GetLocalDateTime());
FSDK_Annot_SetModifiedDateTime(note, GetLocalDateTime());
FSDK_Annot_SetUniqueID(note, WRandomUID());
```

```
// Create a new popup annot and set it to the new note annot.
rectf.left = 300;
rectf.bottom = 450;
rectf.right = 500;
rectf.top = 550;
FS_ANNOT_HANDLE annot1;
FSDK_PDFPage_AddAnnot(page, e_FSPopup, rectf, &annot1)
FS_POPUPANNOT_HANDLE popup;
FSDK_Popup_Create0(annot1, &popup);
FSDK_Annot_SetBorderColor(popup, 0x00FF00);
FSDK_Popup_SetOpenStatus(popup, false);
FSDK_Annot_SetModifiedDateTime(popup, GetLocalDateTime());
FSDK_Markup_SetPopup(note, popup);
...
```

3.14.1.4 How to get a specific annotation in a PDF using device coordinates

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
```

```
#include "include/fs_annot_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
#include "include/fs_pdfpage_c.h"

// Assuming FS_PDFDOC_HANDLE doc has been loaded.
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
...

float fWidth;
float fHeight;
FSDK_PDFPage_GetWidth(page, &fWidth);
FSDK_PDFPage_GetHeight(page, &fHeight);
int width = (int)fWidth;
int height = (int)fHeight;

// Get page transformation matrix.
FSRotation return_Rotation;
FSDK_PDFPage_GetRotation(page, &return_Rotation);
FSMatrix displayMatrix;
FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, return_Rotation, &displayMatrix);
int iAnnotCount;
FSDK_PDFPage_GetAnnotCount(page, &iAnnotCount);

for(int i=0; i<iAnnotCount; i++)
{
    FS_ANNOT_HANDLE pAnnot;
    FSDK_PDFPage_GetAnnot(page, i, &pAnnot);
    FS_BOOL isEmpty;
    FSDK_Annot_IsEmpty(pAnnot, &isEmpty);
    ASSERT_FALSE(isEmpty);
    FSAnnotType type;
    FSDK_Annot_GetType(pAnnot, &type);
    if (e_FSPopup == type) continue;
    FSRectI annotRect;
    FSDK_Annot_GetDeviceRect(pAnnot, displayMatrix, &annotRect);
    FSPointF pt;
    float tolerance = 1.0;

    // Get the same annot (pAnnot) using annotRect.
    pt.x = annotRect.left + tolerance;
    pt.y = (annotRect.top - annotRect.bottom)/2 + annotRect.bottom;
    FS_ANNOT_HANDLE gAnnot;
    FSDK_PDFPage_GetAnnotAtDevicePoint(page, pt, tolerance, &displayMatrix, &gAnnot);
    ...
}
```

3.14.1.5 How to extract the texts under text markup annotations

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
```



```
#include "include/fs_search_c.h"

...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.
...

FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
// Parse the first page.
FS_PROGRESSIVE_HANDLE progressive;
FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
int annot_count;
FSDK_PDFPage_GetAnnotCount(page, &annot_count);
FS_TEXTPAGE_HANDLE text_page;
FSDK_TextPage_Create(page, e_FSTextParseFlagsParseTextNormal, &text_page);
for (int i = 0; i < annot_count; i++) {
    FS_ANNOT_HANDLE annot;
    FSDK_PDFPage_GetAnnot(page, i, &annot);
    FS_TEXTMARKUPANNOT_HANDLE text_markup;
    FSDK_TextMarkup_Create0(annot, &text_markup);
    FS_BOOL return_value;
    FSDK_Annot_IsEmpty(text_markup, &return_value);

    if (!return_value) {
        // Get the texts which intersect with a text markup annotation.
        FS_WSTR text;
        FSDK_TextPage_GetTextUnderAnnot(text_page, text_markup, &text);
    }
}
```

3.14.1.6 How to add richtext for freetext annotation

```
#include "include/fs_common_c.h"
#include "include/fs_annot_c.h"

// Make sure that SDK has already been initialized successfully.
// Load a PDF document, get a PDF page and parse it.

// Add a new freetext annotation, as text box.
FSRectF rectf;
rectf.left = 50;
rectf.bottom = 50;
rectf.right = 150;
rectf.top = 100;

FS_ANNOT_HANDLE annot;
FSDK_PDFPage_AddAnnot(pdf_page, e_FSFreeText, rectf, &annot);
FS_ANNOT_HANDLE freetext;
FSDK_FreeText_Create0(annot, &freetext);
// Set annotation's properties.

// Add/insert richtext string with style.
```

```
FSRichTextStyle richtext_style;
FS_FONT_HANDLE font0;
FSDK_Font_Create(L"Times New Roman", 0, e_FSCharsetANSI, 0, &font0);
richtext_style.font = font0;
richtext_style.text_color = 0xFF0000;
richtext_style.text_size = 10;
richtext_style.text_alignment = e_FSAlignmentLeft;
richtext_style.mark_style = e_FSCornerMarkNone;
richtext_style.is_bold = 0;
richtext_style.is_italic = 0;
richtext_style.is_strikethrough = 0;
richtext_style.is_underline = 0;
FSDK_Markup_AddRichText(freetext, L"Textbox annotation ", richtext_style);

richtext_style.text_color = 0x00FF00;
richtext_style.is_underline = 1;
FSDK_Markup_AddRichText(freetext, L"1-underline ", richtext_style);

FS_FONT_HANDLE font1;
FSDK_Font_Create(L"Calibri", 0, e_FSCharsetANSI, 0, &font1);
richtext_style.font = font1;
richtext_style.text_color = 0x0000FF;
richtext_style.is_underline = 0;
richtext_style.is_strikethrough = 1;
int richtext_count = 0;
FSDK_Markup_GetRichTextCount(freetext, &richtext_count);
FSDK_Markup_InsertRichText(freetext, richtext_count - 1, L"2_strikethrough ", richtext_style);

// Appearance should be reset.
FS_BOOL result = 0;
FSDK_Annot_ResetAppearanceStream(freetext, &result);
FSDK_Annot_Release(freetext);
```

3.14.2 Import annotations from or export annotations to a FDF file

In Foxit PDF SDK, annotations can be created with data not only from applications but also from FDF files. At the same time, PDF SDK supports to export annotations to FDF files.

Example:

3.14.2.1 How to load annotations from a FDF file and add them into the first page of a given PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
#include "include/fs_pdfpage_c.h"
```

```
char* wstring2string(const wchar_t *source, size_t source_size, char *dest, size_t dest_size) {
    char* curLocale = setlocale(LC_ALL, NULL);
    setlocale(LC_ALL, "chs");
    memset(dest, 0, dest_size);
    wcstombs(dest, source, dest_size);
    setlocale(LC_ALL, "C");
    return dest;
}

...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
...
FILE* file = NULL;
_wfopen_s (&file, fdf_file, "rb+");
fseek(file, 0, SEEK_END);
size_t file_size = (size_t)ftell(file);
char* buffer = (char*)malloc(file_size * sizeof(char));
memset(buffer, 0, file_size);

fseek(file, 0, SEEK_SET);
fread(buffer, sizeof(char), file_size, file);
fclose(file);

FS_FDFDOC_HANDLE fdf_doc;
FSDK_FDFDoc_Create2(buffer, file_size, &fdf_doc);
FS_RANGE_HANDLE page_range;
FSDK_Range_Create(&page_range);
FS_BOOL return_result;
FSDK_PDFDoc_ImportFromFDF(pdf_doc, fdf_doc, e_FSAnnots, page_range, &return_result);
```

3.15 Image Conversion

Foxit PDF SDK provides APIs for conversion between PDF files and images. Applications could easily fulfill functionalities like image creation and image conversion which supports the following image formats: BMP, TIFF, PNG, JPX, JPEG, and GIF. Foxit PDF SDK can make the conversion between PDF files and the supported image formats except for GIF. It only supports converting GIF images to PDF files.

Example:

3.15.1 How to convert PDF pages to bitmap files

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_image_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_render_c.h"
```

```
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
...

// Get page count
int nPageCount;
FSDK_PDFDoc_GetPageCount(doc, &nPageCount);
for(int i=0;i<nPageCount;i++) {
    FS_PDFPAGE_HANDLE page;
    FSDK_PDFDoc_GetPage(doc, i, &page);

    // Parse page.
    FS_PROGRESSIVE_HANDLE progressive;
    FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
    float fWidth;
    float fHeight;
    FSDK_PDFPage_GetWidth(page, fWidth);
    FSDK_PDFPage_GetHeight(page, fHeight);
    int width = (int)fWidth;
    int height = (int)fHeight;
    FSMatrix matrix;
    FSRotation return_Rotation;
    FSDK_PDFPage_GetRotation(page, &return_Rotation);
    FSDK_PDFPage_GetDisplayMatrix(page, 0, 0, width, height, return_Rotation, &matrix);

    // Prepare a bitmap for rendering.
    FS_BITMAP_HANDLE bitmap;
    FSDK_Bitmap_Create(width, height, e_FSDIBArgb, NULL, 0, &bitmap);
    FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);

    // Render page.
    FS_RENDERER_HANDLE render;
    FSDK_Renderer_Create(bitmap, false, &render);
    FS_PROGRESSIVE_HANDLE return_StartRender;
    FSDK_Renderer_StartRender(render, page, matrix, NULL, &return_StartRender);
    FS_BOOL return_AddFrame;
    FSDK_Image_AddFrame(image, bitmap, &return_AddFrame);
}
...
```

Note: For pdf2image functionality, if the PDF file contains images larger than 1G, it is recommended to process the images using tiled rendering. Otherwise, it may occur exceptions. Following is a brief implementation of tiled rendering.

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_image_c.h"
#include "include/fs_render_c.h"

...
// Parse page.
```

```
FS_PROGRESSION_HANDLE progressive;
code = FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
float fWidth;
FSDK_PDFPage_GetWidth(page, &fWidth);
int width = (int)(fWidth);
float fHeight;
FSDK_PDFPage_GetHeight(page, &fHeight);
int height = (int)(fHeight);
int render_sum = 10;
int width_scale = 1;
int height_scale = 1;
int little_width = width * width_scale;
int little_height = height / render_sum * height_scale;

for (int i = 0; i < render_sum; i++)
{
    // According to Matrix, do module rendering for large PDF files.
    FSRotation return_rotation;
    FSDK_PDFPage_GetRotation(page, &return_rotation);
    FSMatrix return_matrix;
    FSDK_PDFPage_GetDisplayMatrix(page, 0, -1 * i * little_height, little_width, height * height_scale,
return_rotation, &return_matrix);
    // Prepare a bitmap for rendering.
    FS_BITMAP_HANDLE bitmap;
    FSDK_Bitmap_Create(little_width, little_height, e_FSDIBArgb, NULL, 0, &bitmap);
    FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, NULL);
    // Render page.
    FS_RENDERER_HANDLE render = NULL;
    FSDK_Renderer_Create(bitmap, false, &render);
    FS_PROGRESSION_HANDLE return_StartRender;
    FSDK_Renderer_StartRender(render, page, return_matrix, NULL, &return_StartRender);
    // The bitmap data will be added to the end of image file after rendering.
    ...
}
```

3.15.2 How to convert an image file to PDF file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_image_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"

FS_IMAGE_HANDLE image;
FSDK_Image_Create0(input_file, &image);
int count = 0;
FSDK_Image_GetFrameCount(image, &count);

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create(&doc);
for (int i = 0; i < count; i++) {
    FS_PDFPAGE_HANDLE page;
```

```
int width, height;
FSDK_Image_GetWidth(image, &width);
FSDK_Image_GetHeight(image, &height);
FSDK_PDFDoc_InsertPage(doc, i, width, height, &page);
FS_PROGRESSIVE_HANDLE progressive;
FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
// Add image to page.
FSPointF pointf;
pointf.x = 0;
pointf.y = 0;
FS_BOOL return_result;
FSDK_PDFPage_AddImage(page, image, i, pointf, width, height, true, &return_result);
}

FS_BOOL isSave;
FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &isSave);
...
```

3.16 Watermark

Watermark is a type of PDF annotation and is widely used in PDF document. Watermark is a visible embedded overlay on a document consisting of text, a logo, or a copyright notice. The purpose of a watermark is to identify the work and discourage its unauthorized use. Foxit PDF SDK provides APIs to work with watermark, allowing applications to create, insert, release and remove watermarks.

Example:

3.16.1 How to create a text watermark and insert it into the first page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_watermark_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FSWatermarkSettings settings;
settings.flags = e_FSWatermarkFlagsFlagASPageContents | e_FSWatermarkFlagsFlagOnTop;
settings.offset_x = 0;
settings.offset_y = 0;
settings.opacity = 90;
settings.position = e_FSPosTopRight;
settings.rotation = -45.f;
settings.scale_x = 1.f;
settings.scale_y = 1.f;
FSWatermarkTextProperties text_properties;
text_properties.alignment = e_FSAlignmentCenter;
text_properties.color = 0xF68C21;
```

```
text_properties.font_style = e_FSFontStyleFontStyleNormal;
text_properties.line_space = 1;
text_properties.font_size = 12.f;
FS_FONT_HANDLE font;
FSDK_Font_Create0(e_FSStdIDTimesB, &font);
text_properties.font = font;
FS_WATERMARK_HANDLE watermark;
FSDK_Watermark_Create(doc, L"Foxit PDF SDK\\nwww.foxitsoftware.com", text_properties, settings,
&watermark);
FS_BOOL return_value;
FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
FSDK_Watermark_InsertToPage(watermark, page, &return_value);

// Save document to file
...
```

3.16.2 How to create an image watermark and insert it into the first page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_watermark_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FSWatermarkSettings settings;
settings.flags = e_FSWatermarkFlagsFlagASPageContents | e_FSWatermarkFlagsFlagOnTop;
settings.offset_x = 0.f;
settings.offset_y = 0.f;
settings.opacity = 20;
settings.position = e_FSPosCenter;
settings.rotation = 0.0f;
FS_IMAGE_HANDLE image;
FSDK_Image_Create0(image_file, &image);
FS_BITMAP_HANDLE bitmap;
FSDK_Image_GetFrameBitmap(image, 0, &bitmap);
float Width;
FSDK_PDFPage_GetWidth(page, &Width);
int iWidth;
FSDK_Bitmap_GetWidth(bitmap, &iWidth);
settings.scale_x = Width * 0.618f / iWidth;
settings.scale_y = settings.scale_x;

FS_WATERMARK_HANDLE watermark;
FSDK_Watermark_Create1(doc, image, 0, settings, &watermark);
FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
FS_BOOL return_value;
FSDK_Watermark_InsertToPage(watermark, page, &return_value);
```

```
// Save document to file.
...
```

3.16.3 How to remove all watermarks from a page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"

...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
...
FS_BOOL isRemove;
FSDK_PDFPage_RemoveAllWatermarks(page, &isRemove);
...
// Save document to file
...
```

3.17 Barcode

A barcode is an optical machine-readable representation of data relating to the object to which it is attached. Originally barcodes systematically represented data by varying the widths and spacing of parallel lines, and may be referred to as linear or one-dimensional (1D). Later they evolved into rectangles, dots, hexagons and other geometric patterns in two dimensions (2D). Although 2D systems use a variety of symbols, they are generally referred to as barcodes as well. Barcodes originally were scanned by special optical scanners called barcode readers. Later, scanners and interpretive software became available on devices including desktop printers and smartphones. Foxit PDF SDK provides applications to generate a barcode bitmap from a given string. The barcode types that Foxit PDF SDK supports are listed in Table 3-2.

Table 3-2

Barcode Type	Code39	Code128	EAN 8	UPC A	EAN13	ITF	PDF417	QR
Dimension	1D	1D	1D	1D	1D	1D	2D	2D

Example:

3.17.1 How to generate a barcode bitmap from a string

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_barcode_c.h"
...
```



```
// Strings used as barcode content.
const wchar_t* sz_code_string = L"TEST-SHEET";

// Barcode format types.
FSFormat code_format = e_FSFormatCode39;

//Format error correction level of QR code.
FSQRErrorCorrectionLevel sz_qr_level = e_FSQRCorrectionLevelLow;

//Image names for the saved image files for QR code.
const wchar_t* bmp_qr_name = L"/QR_CODE_TestForBarcodeQrCode_L.bmp";

// Unit width for barcode in pixels, preferred value is 1-5 pixels.
int unit_width = 2;

// Unit height for barcode in pixels, preferred value is >= 20 pixels.
int unit_height = 120;

FS_BARCODE_HANDLE barcode;
FSDK_Barcode_Create(barcode);
FS_BITMAP_HANDLE bitmap;
FSDK_Barcode_GenerateBitmap(barcode, sz_code_string, code_format, unit_width, unit_height, sz_qr_level,
&bitmap);
...
```

3.18 Security

Foxit PDF SDK provides a range of encryption and decryption functions to meet different level of document security protection. Users can use regular password encryption and certificate-driven encryption, or using their own security handler for custom security implementation. It also provides APIs to integrate with the third-party security mechanism (Microsoft RMS). These APIs allow developers to work with the Microsoft RMS SDK to both encrypt (protect) and decrypt (unprotect) PDF documents.

Note: For more detailed information about the RMS encryption and decryption, please refer to the simple demo "**security**" in the "\\examples\\simple_demo" folder of the download package.

Example:

3.18.1 How to encrypt a PDF file with Certificate

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_security_c.h"

...
char* wstring2string(const wchar_t *source, size_t source_size, char *dest, size_t dest_size) {
    char* curLocale = setlocale(LC_ALL, NULL);
```

```

setlocale(LC_ALL, "chs");
memset(dest, 0, dest_size);
wcstombs(dest, source, dest_size);
setlocale(LC_ALL, "C");
return dest;
}
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) {
    FSDK_PDFDoc_Release(doc);
    return false;
}

// Do encryption.
int envelopeslength = 1;
FS_BSTR* envelopes = malloc( envelopeslength * sizeof(FS_BSTR));
FS_BSTR initial_key;
wchar_t cert_file_path[MAX_FILE_PATH];
swprintf_s(cert_file_path, MAX_FILE_PATH, L"%lsfoxit.cer", input_path);
char c_cert_file_path[MAX_FILE_PATH];
wstring2string(cert_file_path, wcslen(cert_file_path), c_cert_file_path, MAX_FILE_PATH)
if (!GetCertificateInfo((const char*)c_cert_file_path, envelopes, envelopeslength, initial_key, true, 16)) {
    return false;
}
FS_CERTIFICATESECURITYHANDLER_HANDLE handler;
FSDK_CertificateSecurityHandler_Create(&handler);
FSCertificateEncryptData encrypt_data;
encrypt_data.cipher = e_FSCipherAES;
encrypt_data.is_encrypt_metadata = true;
encrypt_data.envelopes = envelopes;
encrypt_data.envelopes_length = envelopeslength;
FS_BOOL result;
FSDK_CertificateSecurityHandler_Initialize(handler, encrypt_data, initial_key.str, &result);
free(envelopes);
FSDK_PDFDoc_SetSecurityHandler(doc, handler, &result);
wchar_t output_file[MAX_FILE_PATH];
swprintf_s(output_file, MAX_FILE_PATH, L"%lscertificate_encrypt.pdf", input_path);
FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &result);
...

```

3.18.2 How to encrypt a PDF file with Foxit DRM

```

#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_security_c.h"
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);

```

```

FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) {
    FSDK_PDFDoc_Release(doc);
    return false;
}

// Do encryption.
FS_DRMSECURITYHANDLER_HANDLE handler;
FSDK_DRMSecurityHandler_Create(&handler);
const char* file_id = "Simple-DRM-file-ID";
const char* initialize_key = "Simple-DRM-initialize-key";
FSDRMEncryptData encrypt_data;
encrypt_data.is_encrypt_metadata = true;
encrypt_data.cipher = e_FSCipherAES;
encrypt_data.is_owner = true;
encrypt_data.key_length = 16;
encrypt_data.sub_filter.str = "Simple-DRM-filter";
encrypt_data.sub_filter.len = strlen("Simple-DRM-filter");
encrypt_data.user_permissions = 0xffffffff;
FS_BOOL result;
FSDK_DRMSecurityHandler_Initialize(handler, encrypt_data, file_id, initialize_key &result);
FSDK_PDFDoc_SetSecurityHandler(doc, handler, &result);

wchar_t output_file[MAX_FILE_PATH];
swprintf_s(output_file, MAX_FILE_PATH, L"%lsfoxit_drm_encrypt.pdf", output_directory);
FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &result);
FSDK_PDFDoc_Release(doc);
...

```

3.19 Reflow

Reflow is a function that rearranges page content when the page size changes. It is useful for applications that have output devices with difference sizes. Reflow frees the applications from considering layout for different devices. This function provides APIs to create, render, release and access properties of 'reflow' pages.

Example:

3.19.1 How to create a reflow page and render it to a bmp file

```

#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_reflowpage.h"
#include "include/fs_render_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

```

```
FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
// Parse PDF page.
FS_PROGRESSION_HANDLE progressive;
FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);

FS_REFLOWPAGE_HANDLE reflow_page;
FSDK_ReflowPage_Create(page, &reflow_page);

// Set some arguments used for parsing the reflow page.
FSDK_ReflowPage_SetLineSpace(reflow_page, 0);
FSDK_ReflowPage_SetZoom(reflow_page, 100);
FSDK_ReflowPage_SetParseFlags(reflow_page, e_FSReflowPageFlagsNormal);

// Parse reflow page.
FS_PROGRESSION_HANDLE progressive1;
FSDK_ReflowPage_StartParse(reflow_page, NULL, &progressive1);

// Get actual size of content of reflow page. The content size does not contain the margin.
float content_height;
FSDK_ReflowPage_GetContentHeight(reflow_page, &content_height);
float content_width;
FSDK_ReflowPage_GetContentWidth(reflow_page, &content_width);

// Assuming Bitmap bitmap has been created.

// Render reflow page.
FS_RENDERER_HANDLE renderer;
FSDK_Renderer_Create(bitmap, false, &renderer);
FSMatrix matrix;
FSDK_ReflowPage_GetDisplayMatrix(reflow_page, 0, 0, (int)content_width, (int)content_height, e_FSRotation0,
&matrix);
FS_PROGRESSION_HANDLE progressive2;
FSDK_Renderer_StartRenderReflowPage(renderer, reflow_page, matrix, NULL, &progressive2);
...
```

3.20 Asynchronous PDF

Asynchronous PDF technique is a way to access PDF pages without loading the whole document when it takes a long time. It's especially designed for accessing PDF files on internet. With asynchronous PDF technique, applications do not have to wait for the whole PDF file to be downloaded before accessing it. Applications can open any page when the data of that page is available. It provides a convenient and efficient way for web reading applications. For how to open and parse pages with asynchronous mode, you can refer to the simple demo "**async_load**" in the "\examples\simple_demo" folder of the download package.

3.21 Pressure Sensitive Ink

Pressure Sensitive Ink (PSI) is a technique to obtain varying electrical outputs in response to varying pressure or force applied across a layer of pressure sensitive devices. In PDF, PSI is usually used for hand writing signatures. PSI data are collected by touching screens or handwriting on boards. PSI data contains coordinates and canvas of the operating area which can be used to generate appearance of PSI. Foxit PDF SDK allows applications to create PSI, access properties, operate on ink and canvas, and release PSI.

Example:

3.21.1 How to create a PSI and set the related properties for it

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_psi_c.h"

FS_PSI_HANDLE psi;
FSDK_PSI_Create0(480, 180, true, &psi);

// Set ink diameter.
FSDK_PSI_SetDiameter(psi, 9);

// Set ink color.
FSDK_PSI_SetColor(psi, 0x434236);

// Set ink opacity.
FSDK_PSI_SetOpacity(psi, 0.8f);

// Add points to pressure sensitive ink.
float x = 121.3043f;
float y = 326.6846f;
float pressure = 0.0966f;
FSPointType type = e_FSTypeMoveTo;
FSPointF point;
point.x = x;
point.y = y;
FSDK_PSI_AddPoint(psi, point, type, pressure);
...
```

3.22 Wrapper

Wrapper provides a way for users to save their own data related to a PDF document. For example, when opening an encrypted unauthorized PDF document, users may get an error message. In this case, users can still access wrapper data even when they do not have permissions to the PDF content.

The wrapper data could be used to provide information like where to get decryption method of this document.

Example:

3.22.1 How to open a document including wrapper data

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_render_c.h"

// file_name is PDF document which includes wrapper data.
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
FSDK_PDFDoc_Load(doc_wrapper, NULL);
if (e_FSErrSuccess) {
    FSDK_PDFDoc_Release(doc);
    return false;
}
FS_BOOL return_iswrapper;
FSDK_PDFDoc_IsWrapper(doc, &return_iswrapper);
if (!return_iswrapper){
    return false;
}
FS_INT64 offset;
FSDK_PDFDoc_GetWrapperOffset(doc, &offset);

FileReader file_reader(offset);
file_reader.LoadFile(file_name);
...
```

3.23 PDF Objects

There are eight types of object in PDF: Boolean object, numerical object, string object, name object, array object, dictionary object, stream object and null object. PDF objects are document level objects that are different from page objects (see 3.24) which are associated with a specific page each. Foxit PDF SDK provides APIs to create, modify, retrieve and delete these objects in a document.

Example:

3.23.1 How to remove some properties from catalog dictionary

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfobject_c.h"
...
```

```
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_PDFDICTIONARY_HANDLE catalog;
FSDK_PDFDoc_GetCatalog(doc, &catalog);
if (NULL == catalog) return;

const char* key_strings[] = { "Type", "Boolean", "Name", "String", "Array", "Dict" };
int count = sizeof(key_strings)/sizeof(key_strings[0]);
for (int i = 0; i < count; i++) {
    FS_BOOL return_HasKey;
    FSDK_PDFDictionary_HasKey(catalog, key_strings[i], &return_HasKey);
    if (return_HasKey)
        FSDK_PDFArray_RemoveAt(catalog, &i);
}
...
```

3.24 Page Object

Page object is a feature that allows novice users having limited knowledge of PDF objects (see 3.23 for details of PDF objects) to be able to work with text, path, image, and canvas objects. Foxit PDF SDK provides APIs to add and delete PDF objects in a page and set specific attributes. Using page object, users can create PDF page from object contents. Other possible usages of page object include adding headers and footers to PDF documents, adding an image logo to each page, or generating a template PDF on demand.

Example:

3.24.1 How to create a text object in a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfgraphicsobject_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

FS_POSITION position;
FSDK_GraphicsObjects_GetLastGraphicsObjectPosition(page, e_FSTypeText, &position);
FS_TEXTOBJECT_HANDLE text_object;
FSDK_TextObject_Create(&text_object);

FSDK_GraphicsObject_SetFillColor(text_object, 0xFFFF7F00);

// Prepare text state.
FSTextState state;
state.font_size = 80.0f
```

```
FSDK_Font_Create(L"Simsun", e_FSStylesSmallCap, e_FSCharsetGB2312, 0, &state.font);
state.textmode = e_FSMODEFill;
state.origin_position.x = 0;
state.origin_position.y = 0;
state.wordspace = 0.0f;
state.charspace = 0.0f;
state.textmatrix[0] = 1;
state.textmatrix[1] = 0;
state.textmatrix[2] = 0;
state.textmatrix[3] = 1;
FSDK_TextObject_SetTextState(text_object, page, state, false, 750);

// Set text.
FSDK_TextObject_SetText(text_object, L"Foxit Software");
FS_POSITION last_position;
FSDK_GraphicsObjects_InsertGraphicsObject(page, position, text_object, &last_position);
...
```

3.24.2 How to add an image logo to a PDF page

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfgraphicsobject_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
...

//Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

FS_POSITION position;
FSDK_GraphicsObjects_GetLastGraphicsObjectPosition(page, e_FSTypeImage, &position);
FS_IMAGE_HANDLE image;
FSDK_Image_Create0(image_file, &image);
FS_IMAGEOBJECT_HANDLE image_object;
FS_PDFDOC_HANDLE doc;
FSDK_PDFPage_GetDocument(page, &doc);
FSDK_ImageObject_Create(doc, &image_object);
FSDK_ImageObject_SetImage(image_object, image, 0);

int iwidth;
FSDK_Image_GetWidth(image, &width);
int iheight;
FSDK_Image_GetHeight(image, &height);

float width = (float)iwidth;
float height = (float)iheight;

float page_width;
FSDK_PDFPage_GetWidth(page, &page_width);
float page_height;
FSDK_PDFPage_GetHeight(page, &page_height);
```



```
// Please notice the matrix value.
FSMatrix matrix;
matrix.a = width;
matrix.b = 0;
matrix.c = 0;
matrix.d = height;
matrix.e = (page_width - width) / 2.0f;
matrix.f = (page_height - height) / 2.0f;
FSDK_GraphicsObject_SetMatrix(image_object, &matrix)

FSDK_GraphicsObjects_InsertGraphicsObject(page, position, image_object, &position);
FS_BOOL return_value;
FSDK_GraphicsObjects_GenerateContent(page, &return_value);
...
```

3.25 Marked content

In PDF document, a portion of content can be marked as marked content element. Marked content helps to organize the logical structure information in a PDF document and enables stylized tagged PDF. Tagged PDF has a standard structure types and attributes that allow page content to be extracted and reused for other purposes. More details about marked content could be found in chapter 10.5 of PDF reference 1.7 ^[1].

Example:

3.25.1 How to get marked content in a page and get the tag name

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_image_c.h"
#include "include/fs_pdfgraphicsobject_c.h"
#include "include/fs_pdfobject_c.h"

...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.

FS_POSITION position;
FSDK_GraphicsObjects_GetFirstGraphicsObjectPosition(page, e_FSTypeText, &position);
FS_TEXTOBJECT_HANDLE text_obj;
reinterpret_cast<FS_TEXTOBJECT_HANDLE*>(FSDK_GraphicsObjects_GetGraphicsObject(page, position,
&text_obj));
FS_MARKEDCONTENT_HANDLE content;
FSDK_GraphicsObject_GetMarkedContent(text_obj, &content);
int item_count = 0;
FSDK_MarkedContent_GetItemCount(content, &item_count);

// Get marked content property
```

```

for (int i = 0; i < item_count; i++) {
    FS_BSTR tag_name;
    FSDK_MarkedContent_GetItemTagName(content, i, &tag_name);
    int mcid;
    FSDK_MarkedContent_GetItemMCID(content, i, &mcid);
}
...

```

3.26 Layer

PDF Layers, in other words, Optional Content Groups (OCG), are supported in Foxit PDF SDK. Users can selectively view or hide the contents in different layers of a multi-layer PDF document. Multi-layers are widely used in many application domains such as CAD drawings, maps, layered artwork and multi-language document, etc.

In Foxit PDF SDK, a PDF layer is associated with a layer node. To retrieve a layer node, user should create a PDF [FS_LAYERTREE_HANDLE](#) object first and then call function [FSDK_LayerTree_GetRootNode](#) to get the root layer node of the whole layer tree. Furthermore, you can enumerate all the nodes in the layer tree from the root layer node. Foxit PDF SDK provides APIs to get/set layer data, view or hide the contents in different layers, set layers' name, add or remove layers, and edit layers.

Example:

3.26.1 How to create a PDF layer

```

#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_render_c.h"
#include "include/fs_pdflayer_c.h"

...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.

FS_LAYERTREE_HANDLE layertree;
FSDK_LayerTree_Create(doc, &layertree);
FS_LAYERNODE_HANDLE root;
FSDK_LayerTree_GetRootNode(layertree, &root);
FS_BOOL return_value;
FSDK_LayerTree_IsEmpty(layertree, &return_value);
if (return_value) {
    printf("No layer information!\r\n");
    return;
}
...

```

3.26.2 How to set all the layer nodes information

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_render_c.h"
#include "include/fs_pdflayer_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

void SetAllLayerNodesInformation(LayerNode layer_node) {
    FS_BOOL return_HasLayer;
    FSDK_LayerNode_HasLayer(layer_node, &return_HasLayer);
    if (return_HasLayer) {
        FS_BOOL return_value;
        FSDK_LayerNode_SetDefaultVisible(layer_node, true, &return_value);
        FSDK_LayerNode_SetExportUsage(layer_node, e_FSStateUndefined, &return_value);
        FSDK_LayerNode_SetViewUsage(layer_node, e_FSStateOFF, &return_value);
        FSLayerPrintData print_data;
        print_data.print_state = e_FSStateON;
        print_data.subtype.str = "subtype_print";
        print_data.subtype.len = strlen("subtype_print");
        FSDK_LayerNode_SetPrintUsage(layer_node, print_data, &return_value);
        FSLayerZoomData zoom_data;
        zoom_data.max_factor = 10;
        zoom_data.min_factor = 1;
        FSDK_LayerNode_SetZoomUsage(layer_node, zoom_data, &return_value);
        FS_WSTR return_GetName;
        FSDK_LayerNode_GetName(layer_node, &return_GetName);
        wchar_t new_name[255];
        swprintf_s(new_name, 255, L" [View_OFF_Print_ON_Export_Undefined]%ls", return_GetName.str)
        FSDK_LayerNode_SetName(layer_node, new_name, &return_value);
    }
    int count = 0;
    FSDK_LayerNode_GetChildrenCount(layer_node, &count);
    for (int i = 0; i < count; i++) {
        FS_LAYERNODE_HANDLE child;
        FSDK_LayerNode_GetChild(layer_node, i, &child);
        SetAllLayerNodesInformation(child);
    }
}

FS_LAYERTREE_HANDLE layertree;
FSDK_LayerTree_Create(doc, &layertree);
FS_LAYERNODE_HANDLE root;
FSDK_LayerTree_GetRootNode(layertree, &root);
SetAllLayerNodesInformation(root);
...
```

3.26.3 How to edit layer tree

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_render_c.h"
#include "include/fs_pdflayer_c.h"
...

// Assuming FS_PDFDOC_HANDLE doc has been loaded.

// edit layer tree
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
FSErrorCode code = FSDK_PDFDoc_Load(doc, NULL);
FSDK_LayerTree_Create(doc, &layertree);
FSDK_LayerTree_GetRootNode(layertree, &root);
int children_count = 0;
FSDK_LayerNode_GetChildrenCount(root, &children_count);
FS_BOOL return_value;
FSDK_LayerNode_RemoveChild(root, children_count - 1, &return_value);
FS_LAYERNODE_HANDLE child;
FSDK_LayerNode_GetChild(root, children_count - 2, &child);
FS_LAYERNODE_HANDLE child0;
FSDK_LayerNode_GetChild(root, 0, &child0);
FSDK_LayerNode_MoveTo(child, child0, 0, &return_value);
FS_LAYERNODE_HANDLE return_AddChild;
FSDK_LayerNode_AddChild(root, 0, L"AddedLayerNode", true, &return_AddChild);
FSDK_LayerNode_AddChild(root, 0, L"AddedNode", false, &return_AddChild);
```

3.27 Signature

PDF Signature module can be used to create and sign digital signatures for PDF documents, which protects the security of documents' contents and avoids it to be tampered maliciously. It can let the receiver make sure that the document is released by the signer and the contents of the document are complete and unchanged. Foxit PDF SDK provides APIs to create digital signature, verify the validity of signature, delete existing digital signature, get and set properties of digital signature, display signature and customize the appearance of the signature form fields.

Note: Foxit PDF SDK provides default Signature callbacks which supports the following two types of signature filter and subfilter:

- | | |
|---------------------------|--------------------------------|
| (1) filter: Adobe.PPKLite | subfilter: adbe.pkcs7.detached |
| (2) filter: Adobe.PPKLite | subfilter: adbe.pkcs7.sha1 |

If you use one of the above signature filter and subfilter, you can sign a PDF document and verify the validity of signature by default without needing to register a custom callback.

Example:

3.27.1 How to sign the PDF document with a signature

```
#include "include/fs_basictypes_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_image_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_signature_c.h"

// AdobePPKLiteSignature
const char* filter = "Adobe.PPKLite";
const char* sub_filter = "adbe.pkcs7.detached";

if (!use_default) {
    InitializeOpenSSL();
    sub_filter = "adbe.pkcs7.sha1";
    FSSignatureCallback* sig_callback = (FSSignatureCallback*)malloc(sizeof(FSSignatureCallback));
    sig_callback->user_data = sig_callback;
    sig_callback->StartCalcDigest = gStartCalcDigest;
    sig_callback->ContinueCalcDigest = gContinueCalcDigest;
    sig_callback->GetDigest = gGetDigest;
    sig_callback->Sign = gSign;
    sig_callback->Sign0 = gSign0;
    sig_callback->VerifySigState = gVerifySigState;
    sig_callback->IsNeedPadData = gIsNeedPadData;
    sig_callback->CheckCertificateValidity = gCheckCertificateValidity;
    FS_BOOL return_Callback;
    FSDK_Library_RegisterSignatureCallback(filter, sub_filter, sig_callback, &return_Callback);
}

printf("Use signature callback object for filter \"%s\" and sub-filter \"%s\"\r\n",
       filter, sub_filter);
FS_PDFPAGE_HANDLE pdf_page;
FSDK_PDFDoc_GetPage(pdf_doc, 0, &pdf_page);
// Add a new signature to the first page.
FS_SIGNATURE_HANDLE new_signature = AddSignature(pdf_page, &sub_filter);
// Set filter and subfilter for the new signature.
FSDK_Signature_SetFilter(new_signature, filter);
FSDK_Signature_SetSubFilter(new_signature, sub_filter);
FS_BOOL is_signed;
FSDK_Signature_IsSigned(new_signature, &is_signed);
FS_UINT32 sig_state;
char sig_state_str[256] = { 0 };
FSDK_Signature_GetState(new_signature, &sig_state);
printf("[Before signing] Signed?:%s\t State:%s\r\n",
       is_signed? "true" : "false",
       TransformSignatureStateToString(sig_state_str, 256, sig_state));

// Sign the new signature.
wchar_t signed_pdf_path[MAX_FILE_PATH];
```

```

swprintf_s(signed_pdf_path, MAX_FILE_PATH, L"%lssigned_newssignature.pdf",output_directory);
if (use_default)
    swprintf_s(signed_pdf_path, MAX_FILE_PATH, L"%lssigned_newssignature_default_handler.pdf",output_directory);

wchar_t cert_file_path[MAX_FILE_PATH];
swprintf_s(cert_file_path, MAX_FILE_PATH, L"%lsfoxit_all.pfx", input_path);
const wchar_t* cert_file_password = L"123456";
// Cert file path will be passed back to application through callback function FSSignatureCallback::Sign().
// In this demo, the cert file path will be used for signing in callback function FSSignatureCallback::Sign().
FS_PROGRESSIVE_HANDLE progressive;
FSDK_Signature_StartSign(new_signature, cert_file_path, cert_file_password, e_FSDigestSHA1, signed_pdf_path,
NULL, NULL, &progressive);
printf("[Sign] Finished!\r\n");
FSDK_Signature_IsSigned(new_signature, &is_signed);
FSDK_Signature_GetState(new_signature, &sig_state);
printf("[After signing] Signed?:%s\\tState:%s\\r\\n",
        is_signed? "true" : "false",
        TransformSignatureStateToString(sig_state_str, 256, sig_state));

// Open the signed document and verify the newly added signature (which is the last one).
wprintf(L"Signed PDF file: %ls\\r\\n", signed_pdf_path);
FS_PDFDOC_HANDLE signed_pdf_doc;
FSDK_PDFDoc_Create0(signed_pdf_path, &signed_pdf_doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(signed_pdf_doc, NULL);
if (e_FSErrSuccess !=error_code ) {
    FSDK_PDFDoc_Release(signed_pdf_doc);
    printf("Fail to open the signed PDF file.\\r\\n");
    return;
}
// Get the last signature which is just added and signed.
int sig_count;
FSDK_PDFDoc_GetSignatureCount(signed_pdf_doc, &sig_count);
FS_SIGNATURE_HANDLE signed_signature;
FSDK_PDFDoc_GetSignature(signed_pdf_doc, sig_count - 1, &signed_signature);
// Verify the signature.
FS_PROGRESSIVE_HANDLE progressive2;
FSDK_Signature_StartVerify(signed_signature,NULL ,NULL, &progressive2);
printf("[Verify] Finished!\r\n");
FSDK_Signature_IsSigned(signed_signature, &is_signed);
FSDK_Signature_GetState(signed_signature, &sig_state);
printf("[After verifying] Signed?:%s\\tState:%s\\r\\n",
        is_signed? "true" : "false",
        TransformSignatureStateToString(sig_state_str, 256, sig_state));

```

3.27.2 How to implement signature callback function of signing

```

#include "include/fs_basictypes_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_image_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"

```

```
#include "include/fs_signature_c.h"

// Implementation of pdf::SignatureCallback

// Used for implementing SignatureCallback.
typedef struct _DigestContext
{
    SHA_CTX sha_ctx_;
    FSReaderCallback* file_read_callback_;
    FS_UINT32* byte_range_array_;
    FS_UINT32 byte_range_array_size_;
    FS_BSTR digest_;
    FS_BSTR signed_data;
}DigestContext;

typedef struct _SignatureCallbackData
{
    FS_BSTR sub_filter_;
    DigestContext* digest_context_;
    FS_BSTR cert_file_path_;
    FS_BSTR cert_file_password_;
}SignatureCallbackData;

SignatureCallbackData gsignature_callback_data;

FS_BOOL GetTextFromFile(const DigestContext* digest_context_, unsigned char* file_buffer) {
    FSReaderCallback* file_read;
    FS_UINT32 offset;
    FS_UINT32 size;
    if (!digest_context_ || !digest_context_->file_read_callback_) return FALSE;
    file_read = digest_context_->file_read_callback_;
    offset = digest_context_->byte_range_array_[0];
    size = digest_context_->byte_range_array_[1];
    file_read->ReadBlock(file_read->user_data, file_buffer, digest_context_->byte_range_array_[0],
    digest_context_->byte_range_array_[1]);
    file_read->ReadBlock(file_read->user_data, file_buffer + (digest_context_->byte_range_array_[1] -
    digest_context_->byte_range_array_[0]),
        digest_context_->byte_range_array_[2], digest_context_->byte_range_array_[3]);
    return TRUE;
}

FS_BOOL ParseP12File(const wchar_t* cert_file_path, char* cert_file_password,
    EVP_PKEY** pkey, X509** x509, STACK_OF(X509)** ca) {
    FILE* file = NULL;
    PKCS12* pkcs12;
#ifdef _WIN32 || defined(_WIN64)
    _wfopen_s(&file, cert_file_path, L"rb");
#else
    file = fopen(string::FromUnicode(cert_file_path), "rb");
#endif // defined(_WIN32) || defined(_WIN64)
    if (!file) {
```

```
    return FALSE;
}

pkcs12 = d2i_PKCS12_fp(file, NULL);
fclose(file);
if (!pkcs12) {
    return FALSE;
}

if (!PKCS12_parse(pkcs12, cert_file_password, pkey, x509, ca)) {
    return TRUE;
}

PKCS12_free(pkcs12);
if (!pkey)
    return FALSE;
return TRUE;
}

#define HANDLE_CREATE_TS_ERROR {\
    if (!ret) {\
        S_REQ_free(ts_req);\
        ts_req = NULL;\
        printf("could not create query\n");\
    }\
    TS_MSG_IMPRINT_free(msg_imprint);\
    X509_ALGOR_free(algo);\
    ASN1_OBJECT_free(policy_obj);\
    ASN1_INTEGER_free(nonce_asn1);\
    return ts_req;\
}

ASN1_INTEGER *create_nonce(int bits)
{
    unsigned char buf[20];
    ASN1_INTEGER *nonce = NULL;
    int len = (bits - 1) / 8 + 1;
    int i;

    /* Generating random byte sequence. */
    if (len > (int)sizeof(buf))
    {
        printf("bit count error\n");
        ASN1_INTEGER_free(nonce);
        return NULL;
    }
    if (RAND_bytes(buf, len) <= 0)
    {
        printf("can not generate random number\n");
        ASN1_INTEGER_free(nonce);
        return NULL;
    }
}
```



```
/* Find the first non-zero byte and creating ASN1_INTEGER object. */
for (i = 0; i < len && !buf[i]; ++i);
if (!(nonce = ASN1_INTEGER_new()))
{
    printf("could not create nonce\n");
    ASN1_INTEGER_free(nonce);
    return NULL;
}

OPENSSL_free(nonce->data);
/* Allocate at least one byte. */
nonce->length = len - i;
if (!(nonce->data = (unsigned char *)OPENSSL_malloc(nonce->length + 1)))
{
    printf("out of memory\n");
    ASN1_INTEGER_free(nonce);
    return NULL;
}

memcpy(nonce->data, buf + i, nonce->length);

return nonce;
}

TS_REQ *create_ts_query(unsigned char *digest, int len)
{
    int ret = 0;
    TS_REQ *ts_req = NULL;
    const EVP_MD *md;
    TS_MSG_IMPRINT *msg_imprint = NULL;
    X509_ALGOR *algo = NULL;
    ASN1_OBJECT *policy_obj = NULL;
    ASN1_INTEGER *nonce_asn1 = NULL;

    switch (len) {
    case 20:
        md = EVP_get_digestbyname("sha1");
        break;
    case 32:
        md = EVP_get_digestbyname("sha256");
        break;
    default:
        HANDLE_CREATE_TS_ERROR;
    }

    /* Creating request object. */
    if (!(ts_req = TS_REQ_new()))
        HANDLE_CREATE_TS_ERROR;

    /* Setting version. */
    if (!TS_REQ_set_version(ts_req, 1))
        HANDLE_CREATE_TS_ERROR;
```

```
/* Creating and adding MSG_IMPRINT object. */
if (!(msg_imprint = TS_MSG_IMPRINT_new()))
    HANDLE_CREATE_TS_ERROR;

/* Adding algorithm. */
if (!(algo = X509_ALGOR_new()))
    HANDLE_CREATE_TS_ERROR;
if (!(algo->algorithm = OBJ_nid2obj(EVP_MD_type(md))))
    HANDLE_CREATE_TS_ERROR;
if (!(algo->parameter = ASN1_TYPE_new()))
    HANDLE_CREATE_TS_ERROR;
algo->parameter->type = V_ASN1_NULL;
if (ITS_MSG_IMPRINT_set_algo(msg_imprint, algo))
    HANDLE_CREATE_TS_ERROR;

if (ITS_MSG_IMPRINT_set_msg(msg_imprint, digest, len))
    HANDLE_CREATE_TS_ERROR;

if (ITS_REQ_set_msg_imprint(ts_req, msg_imprint))
    HANDLE_CREATE_TS_ERROR;

/* Setting nonce if requested. */
if (!(nonce_asn1 = create_nonce(64)))
    HANDLE_CREATE_TS_ERROR;
if (nonce_asn1 && ITS_REQ_set_nonce(ts_req, nonce_asn1))
    HANDLE_CREATE_TS_ERROR;

/* Setting certificate request flag if requested. */
if (ITS_REQ_set_cert_req(ts_req, 1))
    HANDLE_CREATE_TS_ERROR;

ret = 1;
HANDLE_CREATE_TS_ERROR;
}

long Get_TS_REP(unsigned char *md, int md_len, unsigned char **pRet)
{
    unsigned char *p;
    unsigned char *tsReq;
    int len;
    char SendBuffer[1024] = { 0 };
    const char* bsAuth = ":";
    FS_BSTR bsDestAuth;
    WSADATA Ws;
    SOCKET ClientSocket;
    unsigned char *totalBuf;
    struct sockaddr_in ServerAddr;
    int Ret = 0;
    int AddrLen = 0;
    HOSTENT * iphost;
```

```
char serverIP[20] = "";
struct servent *pST;
int sendLen;
char recvBuf[8096] = { 0 };
int revtotalLen = 0, recTSL = 0;
int revLen;
int nTSRepPos = 0;
int tsrepl = 0;
char rspCode[4] = { 0 };
char *ptr = NULL;

    TS_REQ *request = create_ts_query(md, md_len);
    len = i2d_TS_REQ(request, NULL);
    tsReq = (unsigned char *)OPENSSL_malloc(len);
    p = tsReq;
    len = i2d_TS_REQ(request, (unsigned char **)&p);
    TS_REQ_free(request);

    FSDK_Codec_Base64Encode(bsAuth, 1*sizeof(char), &bsDestAuth);

sprintf_s(SendBuffer, 1024, "GET %s HTTP/1.1\r\n"
"Accept: */*\r\n"
"Content-Type: application/timestamp-query\r\n"
"Content-Length: %d\r\n"
"Character-Encoding: binary\r\n"
"User-Agent: PPKHandler\r\n"
"Host: %s\r\n"
"Connection: Keep-Alive\r\n"
"Cache-Control: no-cache\r\n"
"Authorization: Basic %s\r\n"
"\r\n",
"/TSAServer.aspx", len, "ca.signfiles.com", bsDestAuth.str);

    totalBuf = (unsigned char *)malloc(len + strlen(SendBuffer));

    memcpy(totalBuf, SendBuffer, strlen(SendBuffer));
    memcpy(totalBuf + strlen(SendBuffer), tsReq, len);
    OPENSSL_free(tsReq);
    WSStartup(MAKEWORD(2, 2), &Ws);
    CilentSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if ((iphost = gethostbyname("ca.signfiles.com")) != NULL)
    {
        int i = 0;
        while (iphost->h_addr_list[i])
        {
            memcpy(&serverIP, inet_ntoa(*(struct in_addr *)iphost->h_addr_list[i]), 20);
            i++;
        }
    }

    pST = getservbyname("http", NULL);
```

```
ServerAddr.sin_family = AF_INET;
ServerAddr.sin_addr.s_addr = inet_addr(serverIP);
ServerAddr.sin_port = pST->s_port;
memset(ServerAddr.sin_zero, 0x00, 8);
connect(CientSocket, (struct sockaddr*)&ServerAddr, sizeof(ServerAddr));
sendLen = send(CientSocket, (const char *)totalBuf, len + (int)strlen(SendBuffer), 0);

free(totalBuf);
revLen = recv(CientSocket, recvBuf, 8096, 0);
if (revLen == 0)
{
    printf("\r\ntsa server unreachable.\r\n");
    closesocket(CientSocket);
    WSACleanup();
    return 0;
}
revtotalLen += revLen;

//get the ts reponse length
strncpy_s(rspCode, 4, recvBuf + 9, 3);

if (strcmp(rspCode, "200") != 0)
{
    printf("\r\n tsa server refuse request.\r\n%s\r\n", recvBuf);
    closesocket(CientSocket);
    WSACleanup();
    return 0;
}

ptr = strstr(recvBuf, "Content-Length") + strlen("Content-Length") + 2;
//nTSRepPos = (int)(strRecv.find("Content-Length") + strlen("Content-Length") + 2);

while (*ptr > 47 && *ptr < 58)
{
    tsrepL = tsrepL * 10 + *ptr - 48;
    ptr += 1;
}

while (recTSL < tsrepL && revtotalLen < tsrepL)
{
    printf("retry, len:%d\n", revLen);
    revLen = recv(CientSocket, recvBuf + revtotalLen, 8096 - revtotalLen, 0);
    recTSL += revLen;
    revtotalLen += revLen;
}
*pRet = (unsigned char *)OPENSSL_malloc(tsrepL);
memcpy(*pRet, recvBuf + revtotalLen - tsrepL, tsrepL);

closesocket(CientSocket);
WSACleanup();

return tsrepL;
```

```
}

int append_tsp_token(PKCS7_SIGNER_INFO *sinfo, unsigned char *ts_rep, long tsrepL)
{
    unsigned char *p = ts_rep;
    TS_RESP *tsp = d2i_TS_RESP(NULL, (const unsigned char**)&p, tsrepL);
    if (tsp != NULL)
    {
        int p7_len;
        unsigned char *p7_der = NULL;
        unsigned char *p = NULL;
        PKCS7* token = TS_RESP_get_token(tsp);
        if (!PKCS7_type_is_signed(token))
        {
            printf("Error in timestamp token: not signed!\n");
            return 1;
        }

        p7_len = i2d_PKCS7(token, NULL);
        p7_der = (unsigned char *)OPENSSL_malloc(p7_len);
        p = p7_der;
        i2d_PKCS7(token, &p);

        if (sinfo)
        {
            //Add timestamp token to the PKCS7 signature object
            ASN1_STRING *value = ASN1_STRING_new();
            ASN1_STRING_set(value, p7_der, p7_len);
            PKCS7_add_attribute(sinfo, NID_id_smime_aa_timeStampToken, V_ASN1_SEQUENCE,
value);
            OPENSSL_free(p7_der);
        }
    } else {
        printf("Error decoding timestamp token!\n");
        return 1;
    }
    return 0;
}

FS_BOOL gStartCalcDigest(void* user_data, const FSReaderCallback* file, const FS_UINT32* byte_range_array,
FS_UINT32 size_of_array, const FS_SIGNATURE_HANDLE signature, const void* client_data) {
    if (gsignature_callback_data.digest_context_) {
        free(gsignature_callback_data.digest_context_);
        gsignature_callback_data.digest_context_ = NULL;
    }
    gsignature_callback_data.digest_context_ = (DigestContext*)malloc(sizeof(DigestContext));
    FSDK_BStr_Init(&gsignature_callback_data.digest_context_->signed_data);
    FSDK_BStr_Init(&gsignature_callback_data.digest_context_->digest_);
    gsignature_callback_data.digest_context_->file_read_callback_ = (FSReaderCallback*)file;
    gsignature_callback_data.digest_context_->byte_range_array_ = (FS_UINT32*)byte_range_array;
```

```
gsignature_callback_data.digest_context->byte_range_array_size_ = size_of_array;

if (!SHA1_Init(&(gsignature_callback_data.digest_context->sha_ctx))) {
    free(gsignature_callback_data.digest_context_);
    gsignature_callback_data.digest_context_ = NULL;
    return FALSE;
}
return TRUE;
}

FSState gContinueCalcDigest(void* user_data, const void* client_data, const FSPauseCallback* pause) {
    FS_UINT32 file_length;
    unsigned char* file_buffer;
    if (!gsignature_callback_data.digest_context_) return e_FSError;

    file_length = gsignature_callback_data.digest_context->byte_range_array[1] +
gsignature_callback_data.digest_context->byte_range_array[3];
    file_buffer = (unsigned char*)malloc(file_length);
    if (!file_buffer || !GetTextFromFile(gsignature_callback_data.digest_context_, file_buffer)) return e_FSError;

    SHA1_Update(&(gsignature_callback_data.digest_context->sha_ctx_), file_buffer, file_length);
    free(file_buffer);
    return e_FSFinished;
}

FS_BSTR gGetDigest(void* user_data, const void* client_data) {
    FS_BSTR digest;
    unsigned char* md;
    if (!gsignature_callback_data.digest_context_) {
        digest.str = NULL;
        digest.len = 0;
        return digest;
    };
    md = (unsigned char*)(OPENSSL_malloc((SHA_DIGEST_LENGTH) * sizeof(unsigned char)));
    if (1 != SHA1_Final(md, &(gsignature_callback_data.digest_context->sha_ctx))) {
        digest.str = NULL;
        digest.len = 0;
        return digest;
    }

    if (gsignature_callback_data.digest_context->digest_.str) {
        FSDK_BStr_Clear(&gsignature_callback_data.digest_context->digest_);
    }

    FSDK_BStr_SetLength(&gsignature_callback_data.digest_context->digest_, SHA_DIGEST_LENGTH);
    FSDK_BStr_Set(&gsignature_callback_data.digest_context->digest_, (const char*)md, SHA_DIGEST_LENGTH);

    OPENSSL_free(md);
    return gsignature_callback_data.digest_context->digest_;
}
```

```
unsigned char* PKCS7Sign(const wchar_t* cert_file_path, char* cert_file_password,
    char* plain_text, int plain_text_size, int* signed_data_size);
FS_BOOL PKCS7VerifySignature(FS_BSTR signed_data, FS_BSTR plain_text);

FS_BSTR gSign(void* user_data, const void* digest, FS_UINT32 digest_length, const wchar_t* cert_path,
    const wchar_t* password, FSDigestAlgorithm digest_algorithm,
    void* client_data) {
    FS_BSTR bstr;
    char* plain_text = NULL;
    int signed_data_length = 0;
    size_t password_size;
    size_t pass_word_size;
    char* pass_word = NULL;
    unsigned char* signed_data_buffer;
    bstr.str = NULL;
    bstr.len = 0;
    if (!signature_callback_data.digest_context_) {
        return bstr;
    }

    if(strcmp("adbe.pkcs7.sha1", gsignature_callback_data.sub_filter_.str) == 0) {
        plain_text = (char*)malloc(digest_length);
        memcpy(plain_text, digest, digest_length);
    }

    password_size = wcslen(password);
    pass_word_size = password_size + 1;
    pass_word = (char*)malloc(pass_word_size);
    signed_data_buffer = PKCS7Sign(cert_path, wstring2string(password, password_size, pass_word,
        pass_word_size),
        plain_text, digest_length, &signed_data_length);
    free(pass_word);
    free(plain_text);
    if (!signed_data_buffer) return bstr;
    if (!signature_callback_data.digest_context_->signed_data.str) {
        FSDK_BStr_Clear(&signature_callback_data.digest_context_->signed_data);
    }

    FSDK_BStr_SetLength(&signature_callback_data.digest_context_->signed_data, signed_data_length);
    FSDK_BStr_Set(&signature_callback_data.digest_context_->signed_data, (char*)(const
        char*)signed_data_buffer, signed_data_length);
    free(signed_data_buffer);
    return signature_callback_data.digest_context_->signed_data;
}

FS_BSTR gSign0(void* user_data, const void* digest, FS_UINT32 digest_length, FSIFX_FileStream*
    cert_file_stream, const wchar_t* password, FSDigestAlgorithm digest_algorithm, void* client_data) {
    FS_BSTR bstr;
    bstr.str = "";
    bstr.len = 0;
    return bstr;
}
```

```
FS_UINT32 gVerifySigState(void* user_data, const void* digest, FS_UINT32 digest_length, const void*
signed_data, FS_UINT32 signed_data_len, void* client_data) {
    // Usually, the content of a signature field is contain the certification of signer.
    // But we can't judge this certification is trusted.
    // For this example, the signer is ourself. So when using api PKCS7_verify to verify,
    // we pass NULL to it's parameter <i>certs</i>.
    // Meanwhile, if application should specify the certificates, we suggest pass flag PKCS7_NOINTERN to
    // api PKCS7_verify.
    FS_BSTR plain_text;
    unsigned char* file_buffer = NULL;
    FS_BSTR signed_data_str;
    FS_BOOL ret;
    if (!gsignature_callback_data.digest_context_) return e_FSStateVerifyErrorData;
    if (strcmp("adbe.pkcs7.sha1", gsignature_callback_data.sub_filter_.str) == 0) {
        FSDK_BStr_Init(&plain_text);
        FSDK_BStr_SetLength(&plain_text, digest_length);
        FSDK_BStr_Set(&plain_text, digest, digest_length);
    } else {
        return e_FSStatesStateUnknown;
    }
    FSDK_BStr_Init(&signed_data_str);
    FSDK_BStr_SetLength(&signed_data_str, signed_data_len);
    FSDK_BStr_Set(&signed_data_str, signed_data, signed_data_len);

    ret = PKCS7VerifySignature(signed_data_str, plain_text);
    if (file_buffer) free(file_buffer);
    FSDK_BStr_Clear(&signed_data_str);
    FSDK_BStr_Clear(&plain_text);
    //this function is only used to verify the intergrity of a signature.
    return ret ? e_FSStateVerifyNoChange : e_FSStateVerifyChange;
}

void gRelease(void* user_data) {
    FSDK_BStr_Clear(&gsignature_callback_data.sub_filter_);
    FSDK_BStr_Clear(&gsignature_callback_data.cert_file_path_);
    FSDK_BStr_Clear(&gsignature_callback_data.cert_file_password_);
    FSDK_BStr_Clear(&gsignature_callback_data.digest_context_>digest_);
    FSDK_BStr_Clear(&gsignature_callback_data.digest_context_>signed_data);

    free(gsignature_callback_data.digest_context_);
    free(user_data);
}

FS_BOOL gIsNeedPadData(void* user_data) { return FALSE; }
FSCertValidity gCheckCertificateValidity(void* user_data, const wchar_t* cert_path, const wchar_t*
cert_password, void* client_data) {
    // User can check the validity of input certificate here.
    // If no need to check, just return e_CertValid.
    return e_FSCertValid;
}
```


3.28 Long term validation (LTV)

Foxit PDF SDK provides APIs to establish long term validation of signatures, which is mainly used to solve the verification problem of signatures that have already expired. LTV requires DSS (Document Security Store) which contains the verification information of the signatures, as well as DTS (Document Timestamp Signature) which belongs to the type of time stamp signature.

In order to support LTV, Foxit PDF SDK provides:

- Support for adding the signatures of time stamp type, and provides a default signature callback for the subfilter "ETSI.RFC3161".
- TimeStampServerMgr and TimeStampServer classes, which are used to set and manager the server for time stamp. The default signature callback for the subfilter "ETSI.RFC3161" will use the default time stamp server.
- LTVVerifier class which offers the functionalities of verifying signatures and adding DSS information to documents. It also provides a basic default RevocationCallback which is required by LTVVerifier.

Following lists an example about how to establish long term validation of signatures using the default signature callback for subfilter "ETSI.RFC3161" and the default RevocationCallback. For more details, please refer to the simple demo "ltv" in the "\examples\simple_demo" folder of the download package.

Example:

3.28.1 How to establish long term validation of signatures using the default signature callback for subfilter "ETSI.RFC3161" and the default RevocationCallback

```
#include "include/fs_basictypes_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_signature_c.h"
#include "include/fs_ltvverifier_c.h"
...

// Initialize time stamp server manager, add and set a default time stamp server, which will be used by default
signature callback for time stamp signature.
FSDK_TimeStampServerMgr_Initialize();
FS_TIMESTAMPERVER_HANDLE timestamp_server;
FS_WSTR password;
password.str = NULL;
password.len = 0;
FSDK_TimeStampServerMgr_AddServer(server_name, server_url, L"", &password, &timestamp_server);
```

```
FSDK_TimeStampServerMgr_SetDefaultServer0(timestamp_server);

// Assume that "signed_pdf_path" represents a signed PDF document which contains signed signature.
FS_PDFDOC_HANDLE pdf_doc;
FSDK_PDFDoc_Create0(signed_pdf_path, &pdf_doc);
FS_PROGRESSION_HANDLE progressive;
FSDK_PDFDoc_StartLoad(pdf_doc, NULL, true, NULL, &progressive);
{
    // Use LTVVerifier to verify and add DSS.
    FS_LTVVERIFIER_HANDLE ltv_verifier;
    FSDK_LTVVerifier_Create(pdf_doc, true, true, false, e_FSSignatureCreationTime, &ltv_verifier);
    // Set verifying mode which is necessary.
    FSDK_LTVVerifier_SetVerifyMode(ltv_verifier, e_FSVerifyModeAcrobat);
    FS_UINT32 length;
    FSDK_LTVVerifier_Verify(ltv_verifier, NULL, &length);
    FS_SIGNATUREVERIFYRESULT_HANDLE*sig_verify_result_array= malloc( length *
sizeof(FS_SIGNATUREVERIFYRESULT_HANDLE));
    FSDK_LTVVerifier_Verify(ltv_verifier, sig_verify_result_array, &length);
    for (size_t i = 0; i < length; i++) {
        // ltv state would be e_LTVStateNotEnable here.
        FS_SIGNATUREVERIFYRESULT_HANDLE sig_verify_result = sig_verify_result_array[i];
        FS_LTVState ltv_state;
        FSDK_SignatureVerifyResult_GetLTVState(sig_verify_result, &ltv_state);
        FS_UINT32 sig_state;
        FSDK_SignatureVerifyResult_GetSignatureState(sig_verify_result, &sig_state);
        if (sig_state & e_FSStateVerifyValid)
            FSDK_LTVVerifier_AddDSS(ltv_verifier, sig_verify_result_array[i]);
    }
    free(sig_verify_result_array);
}

// Add a time stamp signature as DTS and sign it. "saved_ltv_pdf_path" represents the newly saved signed PDF
file.
FS_PDFPAGE_HANDLE pdf_page;
FSDK_PDFDoc_GetPage(pdf_doc, 0, &pdf_page);
// The new time stamp signature will have default filter name "Adobe.PPKLite" and default subfilter name
"ETSI.RFC3161".
FS_SIGNATURE_HANDLE timestamp_signature;
FSRectF rect;
FSDK_PDFPage_AddSignature1(pdf_page, rect, L"", e_FSSignatureTypeTimeStamp, true, &timestamp_signature);
FS_PROGRESSION_HANDLE sign_progressive;
FSDK_Signature_StartSign(timestamp_signature, L"", L"", e_FSDigestSHA256, saved_ltv_pdf_path, NULL, NULL,
&sign_progressive);
int rate;
FSDK_Progressive_GetRateOfProgress(sign_progressive, &rate);
if (rate != 100){
    FSState state;
    FSDK_Progressive_Continue(sign_progressive, &state);
}
// Then use LTVVeirfier to verify the new signed PDF file.
FS_PDFDOC_HANDLE check_pdf_doc;
FSDK_PDFDoc_Create0(saved_ltv_pdf_path, &check_pdf_doc);
```

```
FS_PROGRESSIVE_HANDLE progressive1;
FSDK_PDFDoc_StartLoad(check_pdf_doc, NULL, true, NULL, &progressive1);
{
    // Use LTVVerfier to verify.
    FS_LTVVERIFIER_HANDLE ltv_verifier;
    FSDK_LTVVerifier_Create(pdf_doc, true, true, false, e_FSSignatureCreationTime, &ltv_verifier);
    // Set verifying mode which is necessary.
    FSDK_LTVVerifier_SetVerifyMode(ltv_verifier, e_FSVerifyModeAcrobat);
    FS_UINT32 length;
    FSDK_LTVVerifier_Verify(ltv_verifier, NULL, &length);
    FS_SIGNATUREVERIFYRESULT_HANDLE*sig_verify_result_array= malloc (length *
sizeof(FS_SIGNATUREVERIFYRESULT_HANDLE));
    FSDK_LTVVerifier_Verify(ltv_verifier, sig_verify_result_array, &length);
    for (size_t i = 0; i < length; i++) {
        // ltv state would be e_LTVStateEnable here.
        FS_SIGNATUREVERIFYRESULT_HANDLE sig_verify_result = sig_verify_result_array[i];
        FSLTVState ltv_state;
        FSDK_SignatureVerifyResult_GetLTVState(sig_verify_result, &ltv_state);
        ... // User can get other information from SignatureVerifyResult.
    }
    free(sig_verify_result_array);
}

// Release time stamp server manager when everything is done.
FSDK_TimeStampServerMgr_Release();
```

3.29 PAdES

Foxit PDF SDK also supports PAdES (PDF Advanced Electronic Signature) which is the application for CAdES signature in the field of PDF. CAdES is a new standard for advanced digital signature, its default subfilter is "ETSI.CAdES.detached". PAdES signature includes four levels: B-B, B-T, B-LT, and B-LTA.

- B-B: Must include the basic attributes.
- B-T: Must include document time stamp or signature time stamp to provide trusted time for existing signatures, based on B-B.
- B-LT: Must include DSS/VRI to provide certificates and revocation information, based on B-T.
- B-LTA: Must include the trusted time DTS for existing revocation information, based on B-LT.

Foxit PDF SDK provides a default signature callback for the subfilter "ETSI.CAdES.detached" to sign and verify the signatures (with subfilter "ETSI.CAdES.detached"). It also provides TimeStampServerMgr and TimeStampServer classes to set and manager the server for time stamp. The default signature callback for the subfilter "ETSI.CAdES.detached" will use the default time stamp server.

Foxit PDF SDK provides functions to get the level of PAdES from signature, and application level can also judge and determine the level of PAdES according to the requirements of each level. For more details about how to add, sign and verify a PAdES signature in PDF document, please refer to the simple demo "**pades**" in the "\examples\simple_demo" folder of the download package.

3.30 PDF Action

PDF Action is represented as the base PDF action class. Foxit PDF SDK provides APIs to create a series of actions and get the action handlers, such as embedded goto action, JavaScript action, named action and launch action, etc.

Example:

3.30.1 How to create a URI action and insert to a link annot

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfobject_c.h"

// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
// Assuming the annots in the page have been loaded.
...

// Add link annotation
FS_LINKANNOT_HANDLE link;
FSRectF rectf;
rectf.left = 350;
rectf.bottom = 350;
rectf.right = 380;
rectf.top = 400;
FSDK_PDFPage_AddAnnot(page, e_FSLink, rectf, &link);
FSDK_Link_SetHighlightingMode(link, e_FSHighlightingToggle);
// Add action for link annotation
FS_PDFDOC_HANDLE doc;
FSDK_PDFPage_GetDocument(page, &doc);
FS_ACTION_HANDLE action;
FSDK_Action_Create(doc, e_FSTypeURI, &action);
FS_URIACTION_HANDLE uriaction;
FSDK_URIAction_Create(action, &uriaction);
FSDK_URIAction_SetTrackPositionFlag(uriaction, true);
FSDK_URIAction_SetURI(uriaction, "www.foxitsoftware.com");
FSDK_Link_SetAction(link, uriaction);

// Appearance should be reset.
FS_BOOL result;
FSDK_Annot_ResetAppearanceStream(link, &result);
```

```
FSDK_Action_Release(uriaction);  
FSDK_Link_Release(link);  
FSDK_Annot_Release(annot);  
FSDK_PDFDoc_Release(doc);
```

3.30.2 How to create a GoTo action and insert to a link annot

```
#include "include/fs_basictypes_c.h"  
#include "include/fs_common_c.h"  
#include "include/fs_action_c.h"  
#include "include/fs_annot_c.h"  
#include "include/fs_pdfdoc_c.h"  
#include "include/fs_pdfobject_c.h"  
#include "include/fs_pdfpage_c.h"  
  
// Assuming the FS_PDFDOC_HANDLE doc has been loaded.  
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.  
  
// Add link annotation  
FS_LINKANNOT_HANDLE link;  
FSRectF rectf;  
rectf.left = 350;  
rectf.bottom = 350;  
rectf.right = 380;  
rectf.top = 400;  
FSDK_PDFPage_AddAnnot(page, e_FSLink, rectf, &link);  
FSDK_Link_SetHighlightingMode(link, e_FSHighlightingToggle);  
  
FS_GOTOACTION_HANDLE gotoaction;  
FS_PDFDOC_HANDLE doc;  
FSDK_PDFPage_GetDocument(page, &doc);  
FS_ACTION_HANDLE action;  
FSDK_Action_Create(doc, e_FSTypeGoto, &action);  
FSDK_GotoAction_Create(gotoaction, &action);  
FS_DESTINATION_HANDLE newDest;  
FSDK_Destination_CreateXYZ(doc, 0, 0, 0, 0, &newDest);  
FSDK_GotoAction_SetDestination(gotoaction, newDest);
```

3.31 JavaScript

JavaScript was created to offload Web page processing from a server onto a client in Web-based applications. Foxit PDF SDK JavaScript implements extensions, in the form of new objects and their accompanying methods and properties, to the JavaScript language. It enables a developer to manage document security, communicate with a database, handle file attachments, and manipulate a PDF file so that it behaves as an interactive, web-enabled form, and so on.

JavaScript action is an action that causes a script to be compiled and executed by the JavaScript interpreter.

The JavaScript methods and properties supported by Foxit PDF SDK are listed in the [appendix](#).

Example:

3.31.1 How to add JavaScript Action to Document

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"

// Load Document doc.
...

FS_JAVASCRIPTACTION_HANDLE javascript_action;
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action, L"app.alert(\"Hello Foxit \");");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create(doc, NULL, &additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerDocWillClose, javascript_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerDocWillClose, &result);
...
```

3.31.2 How to add JavaScript Action to Annotation

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"

// Load Document and get a widget annotation.
...

FS_JAVASCRIPTACTION_HANDLE javascript_action;
FS_PDFDOC_HANDLE doc;
FSDK_PDFPage_GetDocument(page, &doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action, L"app.alert(\"Hello Foxit \");");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create2(annot, &additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerAnnotMouseButtonPressed, javascript_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotMouseButtonPressed, &result);
...
```

3.31.3 How to add JavaScript Action to FormField

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"
#include "include/fs_pdfform_c.h"

// Load Document and get a form field.
```

...

// Add text field.

FS_CONTROL_HANDLE control;

FSRectF rectf;

rectf.left = 50;

rectf.bottom = 600;

rectf.right = 90;

rectf.top = 640;

FSDK_Form_AddControl(form, page, L"Text Field0", e_FSTypeTextField, rectf, &control);

FS_FIELD_HANDLE field;

FSDK_Control_GetField(control, &field);

FSDK_Field_SetValue(field, L"3");

// Update text field's appearance.

FS_WIDGETANNOT_HANDLE widget;

FSDK_Control_GetWidget(control, &widget);

FS_BOOL result;

FSDK_Annot_ResetAppearanceStream(widget, &result);

FS_CONTROL_HANDLE control1;

rectf.left = 100;

rectf.bottom = 600;

rectf.right = 140;

rectf.top = 640;

FSDK_Form_AddControl(form, page, L"Text Field1", e_FSTypeTextField, rectf, &control1);

FS_FIELD_HANDLE field1;

FSDK_Control_GetField(control1, &field1);

FSDK_Field_SetValue(field1, L"23");

// Update text field's appearance.

FS_WIDGETANNOT_HANDLE widget1;

FSDK_Control_GetWidget(control1, &widget1);

FS_BOOL result1;

FSDK_Annot_ResetAppearanceStream(widget1, &result1);

FS_CONTROL_HANDLE control2;

rectf.left = 150;

rectf.bottom = 600;

rectf.right = 190;

rectf.top = 640;

FSDK_Form_AddControl(form, page, L"Text Field2", e_FSTypeTextField, rectf, &control2);

FS_JAVASCRIPTACTION_HANDLE javascript_action;

FS_PDFDOC_HANDLE doc;

FSDK_Form_GetDocument(form, &doc);

(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);

FSDK_JavaScriptAction_SetScript(javascript_action, L"AFSimple_Calculate(\"SUM\", new Array (\"Text Field0\", \"Text Field1\"));");

FS_FIELD_HANDLE field2;

FSDK_Control_GetField(control2, &field2);

FS_ADDITIONALACTION_HANDLE additional_act;

FSDK_AdditionalAction_Create1(field2, &additional_act);

FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerFieldRecalculateValue, javascript_action);

// Update text field's appearance.

```
FS_WIDGETANNOT_HANDLE widget2;
FSDK_Control_GetWidget(control2, &widget2);
FS_BOOL result2;
FSDK_Annot_ResetAppearanceStream(widget2, &result2);
...
```

3.31.4 How to add a new annotation to PDF using JavaScript

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"

// Load Document and get form field, construct a Form object and a Filler object.
...

FS_JAVASCRIPTACTION_HANDLE javascript_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, &doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action, L"var annot = this.addAnnot({ page : 0, type : \"Square\", rect :
[ 0, 0, 100, 100 ], name : \"UniqueID\", author : \"A. C. Robot\", contents : \"This section needs revision.\" });");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create1(field, &additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerAnnotCursorEnter, javascript_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotCursorEnter, &result);
...
```

3.31.5 How to get/set properties of annotations (strokeColor, fillColor, readOnly, rect, type) using JavaScript

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"

// Load Document and get form field, construct a Form object and a Filler object.
...

// Get properties of annotations.
FS_JAVASCRIPTACTION_HANDLE javascript_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, &doc);
FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action, L"var ann = this.getAnnot(0, \" UniqueID \"); if (ann != null)
{ console.println(\"Found it! type: \" + ann.type); console.println(\"readOnly: \" + ann.readOnly);
console.println(\"strokeColor: \" + ann.strokeColor);console.println(\"fillColor: \" + ann.fillColor);
console.println(\"rect: \" + ann.rect);}");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create1(field, &additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerAnnotCursorEnter, javascript_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotCursorEnter, &result);
```



```
// Set properties of annotations (only take strokeColor as an example).
FS_JAVASCRIPTACTION_HANDLE javascript_action1 ;
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action1, L"var ann = this.getAnnot(0, \"UniqueID\");if (ann != null)
{ ann.strokeColor = color.blue; }");
FS_ADDITIONALACTION_HANDLE additional_act1;
FSDK_AdditionalAction_Create1(field1, &additional_act1);
FSDK_AdditionalAction_SetAction(additional_act1, e_FSTriggerAnnotCursorEnter, javascript_action);
FS_BOOL result1;
FSDK_AdditionalAction_DoJSAction(additional_act1, e_FSTriggerAnnotCursorEnter, &result1);
...
```

3.31.6 How to destroy annotation using JavaScript

```
#include "include/fs_basictypes_c.h"
#include "include/fs_action_c.h"
#include "include/fs_annot_c.h"

// Load Document and get form field, construct a Form object and a Filler object.
...

FS_JAVASCRIPTACTION_HANDLE javascript_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, &doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, &javascript_action);
FSDK_JavaScriptAction_SetScript(javascript_action, L"var ann = this.getAnnot(0, \" UniqueID \"); if (ann != null)
{ ann.destroy(); } ");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create1(field, &additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerAnnotCursorEnter, &javascript_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotCursorEnter, &result);
...
```

3.32 Redaction

Redaction is the process of removing sensitive information while keeping the document's layout. It allows users to permanently remove (redact) visible text and images from PDF documents to protect confidential information, such as social security numbers, credit card information, product release dates, and so on.

Redaction is a type of markup annotation, which is used to mark some contents of a PDF file and then the contents will be removed once the redact annotations are applied.

To do Redaction, you can use the following APIs:

- Call function [FSDK_Redaction_Create](#) to create a redaction module. If module "Redaction" is not defined in the license information which is used in function [FSDK_Library_Initialize](#), it

means user has no right in using redaction related functions and this constructor will throw exception `e_FSInvalidLicense`.

- Then call function `FSDK_Redaction_MarkRedactAnnot` to create a redaction object and mark page contents (text object, image object, and path object) which are to be redacted.
- Finally call function `FSDK_Redaction_Apply` to apply redaction in marked areas: remove the text or graphics under marked areas permanently.

Note: To use the redaction feature, please make sure the license key has the permission of the 'Redaction' module.

Example:

3.32.1 How to redact the text "PDF" on the first page of a PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_search_c.h"
#include "include/fs_redaction_c.h"
#include "include/fs_render_c.h"
...

FS_REDACTION_HANDLE redaction;
FSDK_Redaction_Create(doc, &redaction);
// Parse PDF page.
FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(doc, 0, &page);
FS_PROGRESSION_HANDLE progressive;
FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
//Find Text Object to redact
FS_TEXTPAGE_HANDLE text_page;
FSDK_TextPage_Create(page, 0, &text_page);
FS_TEXTSEARCH_HANDLE text_search;
FSDK_TextSearch_Create1(text_page, &text_search);
FS_BOOL return_code;
FSDK_TextSearch_SetPattern(text_search, L"PDF", &return_code);
FSRectF* rect_array = NULL;
FS_BOOL return_FindNext;
FSDK_TextSearch_FindNext(text_search, return_FindNext);
while(return_FindNext) {
    FSRectF* rect = NULL;
    FS_UINT32 return_array_length = 0;
    FSDK_TextSearch_GetMatchRects(text_search, rect, &return_array_length);
    rect = malloc( return_array_length * sizeof(FSRectF));
    FSDK_TextSearch_GetMatchRects(text_search, rect, &return_array_length);
    for (int i = 0; i < return_array_length; i++) {
        FSRectF rectF = rect[i];
```

```
    rectarray = (FSRectF*)realloc(rectarray, count * sizeof(FSRectF));
    rectarray[count - 1] = rect[i];
}
free(rect);
FSDK_TextSearch_FindNext(text_search, &return_FindNext);
}
if(rect_array.GetSize() > 0) {
    FSRectF* rect = malloc(rectarray.size() * sizeof(FSRectF));
    for (int i = 0; i < rectarray.size(); i++)
        rect[i] = rect_array[i];
    FS_REDACTION_HANDLE redact;
    FSDK_Redaction_MarkRedactAnnot(redaction, page, rect, rect_array.size(), &redact);
    free(rect);
    FS_BOOL return_code;
    FSDK_Annot_ResetAppearanceStream(redact, &return_code);
    wchar_t save_pdf_path[MAX_FILE_PATH] = { 0 };
    swprintf_s(save_pdf_path, MAX_FILE_PATH, L"%lsAboutFoxit_redacted_default.pdf", output_directory);
    FS_PROGRESSIVE_HANDLE return_StartSaveAs1;
    FSDK_PDFDoc_StartSaveAs(doc, save_pdf_path, e_FSSaveFlagsSaveFlagNormal, NULL,
&return_StartSaveAs1);

    // set border color to Green
    FSDK_Annot_SetBorderColor(redact, (long)0x00FF00);
    // set fill color to Blue
    FSDK_Redact_SetFillColor(redact, (long)0x0000FF);
    // set rollover fill color to Red
    FSDK_Redact_SetApplyFillColor(redact, (long)0xFF0000);
    FSDK_Annot_ResetAppearanceStream(redact, &return_code);
    swprintf_s(save_pdf_path1, MAX_FILE_PATH, L"%ls AboutFoxit_redacted_setColor.pdf",
output_directory);
    FS_PROGRESSIVE_HANDLE return_StartSaveAs2;
    FSDK_PDFDoc_StartSaveAs(doc, save_pdf_path1, e_FSSaveFlagsSaveFlagNormal, NULL,
&return_StartSaveAs2);

    FSDK_Markup_SetOpacity(redact, (float)0.5);
    FSDK_Annot_ResetAppearanceStream(redact, &return_code);
    wchar_t save_pdf_path2[MAX_FILE_PATH];
    swprintf_s(save_pdf_path2, MAX_FILE_PATH, L"%lsAboutFoxit_redacted_setOpacity.pdf",
output_directory);
    FS_PROGRESSIVE_HANDLE return_StartSaveAs3;
    FSDK_PDFDoc_StartSaveAs(doc, save_pdf_path2, e_FSSaveFlagsSaveFlagNormal, NULL,
&return_StartSaveAs3);
    FSDK_Redaction_Apply(redaction, &return_code);
    if(return_code)
        printf("Redact page(0) succeed.\n");
    else
        printf("Redact page(0) failed.\n");
}
swprintf_s(save_pdf_path3, MAX_FILE_PATH, L"%lsAboutFoxit_redacted_apply.pdf", output_directory);
FS_PROGRESSIVE_HANDLE return_StartSaveAs;
FSDK_PDFDoc_StartSaveAs(doc, save_pdf_path3, e_FSSaveFlagsSaveFlagNormal, NULL, &return_StartSaveAs);
```

3.33 Comparison

Comparison feature lets you see the differences in two versions of a PDF. Foxit PDF SDK provides APIs to compare two PDF documents page by page, the differences between the two documents will be returned.

The differences can be defined into three types: delete, insert and replace. You can save these differences into a PDF file and mark them as annotations.

Note: To use the comparison feature, please make sure the license key has the permission of the 'Comparison' module

Example:

3.33.1 How to compare two PDF documents and save the differences between them into a PDF file

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_search_c.h"
#include "include/fs_compare_c.h"
#include "include/fx_basic_c.h"

...

FS_PDFDOC_HANDLE base_doc;
FSDK_PDFDoc_Create0(input_base_file, &base_doc);
FSDK_PDFDoc_Load(base_doc, NULL);
if (error_code != e_FSErrSuccess) {
    FSDK_PDFDoc_Release(base_doc);
    return 1;
}

FS_PDFDOC_HANDLE compared_doc;
FSDK_PDFDoc_Create0(input_compared_file, &compared_doc);
error_code = FSDK_PDFDoc_Load(compared_doc, NULL);
if (error_code != foxit::e_ErrSuccess) {
    FSDK_PDFDoc_Release(compared_doc);
    return 1;
}

FS_COMPARISON_HANDLE comparison;
FSDK_Comparison_Create(base_doc, compared_doc, &comparison);

// Start comparing.
FSCmpareResults result;
result.base_doc_results_array = NULL;
```

```

result.compared_doc_results_array = NULL;
FSDK_Comparison_DoCompare(comparison, 0, 0, e_FSCompareTypeText, &result);
result.base_doc_results_array = malloc(sizeof(FSCompareResultInfo) * result.base_doc_results_array_length);
for (int i = 0; i < result.base_doc_results_array_length; i++)
    result.base_doc_results_array[i].rect_array = NULL;

result.compared_doc_results_array = malloc( sizeof(FSCompareResultInfo) *
result.compared_doc_results_array_length);
for (int i = 0; i < result.compared_doc_results_array_length; i++)
    result.compared_doc_results_array[i].rect_array = NULL;

FSDK_Comparison_DoCompare(comparison, 0, 0, e_FSCompareTypeText, &result);
for (int i = 0; i < result.base_doc_results_array_length; i++)
    result.base_doc_results_array = malloc(sizeof(FSCompareResultInfo) *
result.compared_doc_results_array[i].array_length_rect_array);
for (int i = 0; i < result.compared_doc_results_array_length; i++)
    result.compared_doc_results_array[i].rect_array = malloc(sizeof(FSCompareResultInfo) *
result.base_doc_results_array_length);
FSDK_Comparison_DoCompare(comparison, 0, 0, e_FSCompareTypeText, &result);

FS_PDFPAGE_HANDLE page;
FSDK_PDFDoc_GetPage(compared_doc, 0, &page);
for (int i=0; i<result.compared_doc_results_array_length; i++)
{
    const FSCompareResultInfo item = result.compared_doc_results_array[i];
    FSCompareResultType type = item.type;
    if (type == e_FSCompareResultTypeDeleteText)
    {
        wchar_t res_string_new[100];
        swprintf_s(res_string_new, L"%ls", item.diff_contents.str);

        // Add stamp to mark the "delete" type differences between the two documents.
        CreateDeleteTextStamp(page, item.rect_array, item.array_length_rect_array, 0xff0000, res_string_new,
L"Compare : Delete", L"Text");
    }
    else if (type == e_FSCompareResultTypeInsertText)
    {
        wchar_t res_string_new[100];
        swprintf_s(res_string_new, 100, L"%ls", item.diff_contents.str);

        // Highlight the "insert" type differences between the two documents.
        CreateDeleteText(page, item.rect_array, item.array_length_rect_array, 0x0000ff, res_string_new,
L"Compare : Insert", L"Text");
    }
    else if (type == e_FSCompareResultTypeReplaceText)
    {
        wchar_t res_string_new[100];
        swprintf_s(res_string_new, 100, "[Old]: \"%s\\r\\n[New]: \"%s\\",
result.base_doc_results_array[i].diff_contents.str, item.diff_contents.str);
        // Squiggly the "replace" type differences between the two documents.
        CreateSquigglyRect(page, item.rect_array, item.array_length_rect_array, 0xe7651a, res_string_new,
L"Compare : Replace", L"Text");
    }
}

```

```
}  
}  
  
// Save the comparison result to a PDF file.  
FS_BOOL return_result = false;  
  
wchar_t output_compared_doc[MAX_FILE_PATH];  
  
swprintf_s(output_compared_doc, MAX_FILE_PATH, L"%lsnew.pdf", output_directory);  
FSDK_PDFDoc_SaveAs(compared_doc, output_compared_doc, e_FSSaveFlagsSaveFlagNormal, &return_result);
```

Note: for *CreateDeleteTextStamp*, *CreateDeleteText* and *CreateSquigglyRect* functions, please refer to the simple demo "**pdfcompare**" located in the "\\examples\\simple_demo" folder of the download package.

3.34 OCR

Optical Character Recognition, or OCR, is a software process that enables images or printed text to be translated into machine-readable text. OCR is most commonly used when scanning paper documents to create electronic copies, but can also be performed on existing electronic documents (e.g. PDF).

From version 9.0, Linux x64 platform supports OCR feature, and the OCR engine has been upgraded, please contact Foxit support team or sales team to get the latest engine files package.

This section will provide instructions on how to set up your environment for the OCR feature module using Foxit PDF SDK for Windows (C API).

3.34.1 System requirements

Platform: Windows, Linux (x64)

Programming Language: C, C++, Java, Python, C#, Node.js

License Key requirement: 'OCR' module permission in the license key

SDK Version: Foxit PDF SDK for Windows (C++, Java, C#) 6.4 or higher; Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK for Windows (Python) 8.3 or higher; Foxit PDF SDK for Linux x64 (C++, Java, C#, Python) 9.0 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.34.2 Trial limit for SDK OCR add-on module

For the trial version, there are three trial limits that you should notice:

- 1) Allow 30 consecutive natural days to evaluate SDK from the first time of OCREngine initialization.
- 2) Allow up to 5000 pages to be converted using OCR from the first time of OCREngine initialization.

- 3) Trail watermarks will be generated on the PDF pages. This limit is used for all of the SDK modules.

3.34.3 OCR resource files

Please contact Foxit support team or sales team to get the OCR resource files package.

After getting the package, extract it to a desired directory (for example, extract the package to a directory named "ocr_addon"), and then you can see the resource files for OCR are as follows:

- **debugging_files:** Resource files used for debugging the OCR project. These file(s) cannot be distributed.
- **language_resource_CJK:** Resource files for CJK language, including: Chinese-Simplified, Chinese-Traditional, Japanese, and Korean.
- **language_resources_noCJK:** Resource files for the languages except CJK, including: Basque, Bulgarian, Catalan, Croatian, Czech, Danish, Dutch, English, Estonian, Faeroese, Finnish, French, Galician, German, Greek, Hebrew, Hungarian, Icelandic, Italian, Latvian(Lettish), Lithuanian, Macedonian, Maltese, Norwegian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak, Slovenian, Spanish, Swedish, Thai, Turkish, Ukrainian.
- **win32_lib:** 32-bit library resource files
- **win64_lib:** 64-bit library resource files
- **readme.txt:** A txt file for introducing the role of each folder in this directory, as well as how to use those resource files for OCR.

3.34.4 How to run the OCR demo

Foxit PDF SDK for C API provides an OCR demo located in the "\examples\simple_demo\ocr" folder to show you how to use Foxit PDF SDK to do OCR for a PDF page or a PDF document.

3.34.4.1 Build an OCR resource directory

Before running the OCR demo, you should first build an OCR resource directory, and then pass the directory to Foxit PDF SDK API **FSDK_OCREngine_Initialize** to initialize OCR engine.

Note: Starting from version 10.1, Foxit PDF SDK has added a new interface naming **FSDK_OCREngine_Initialize0**, and added a new parameter, **is_shared_cpu_cores_mode**, to specify whether to use multi-process mode. When this parameter value is true, multi-process mode will be used during the OCR process, and when the value is false, single-process mode will be used.

To build an OCR resource directory, please follow the steps below:

- 1) Create a new folder to add the resources. For example, "D:/ocr_resources".
- 2) Add the appropriate library resource based on the platform architecture.
 - For **win32**, copy **all the files** under "ocr_addon/win32_lib" folder to "D:/ocr_resources".
 - For **win64**, copy **all the files** under "ocr_addon/win64_lib" folder to "D:/ocr_resources".
- 3) Add the language resource.
 - For CJK (Chinese-Simplified, Chinese-Traditional, Japanese, and Korean), copy **all the files** under "ocr_addon/language_resource_CJK" folder to "D:/ocr_resources".
 - For all other languages except CJK, copy **all the files** under "ocr_addon/language_resources_noCJK" folder to "D:/ocr_resources".
 - For all the supported languages, copy **all the files** under "ocr_addon/language_resource_CJK" and "ocr_addon/language_resources_noCJK" folders to "D:/ocr_resources".
- 4) (Optional) Add debugging file resource if you need to debug the demo.
 - For win32, copy the file(s) under "ocr_addon/debugging_files/win32" folder to "D:/ocr_resources".
 - For win64, copy the file(s) under "ocr_addon/debugging_files/win64" folder to "D:/ocr_resources".

Note: The debugging files should be exclusively used for testing purposes. So, you cannot distribute them.

3.34.4.2 Configure the demo

After building the OCR resource directory, configure the demo in the "examples\simple_demo\ocr\ocr.c" file.

Specify the OCR resource directory

Add the OCR resource directory as follows, which will be used to initialize the OCR engine.

```
// "ocr_resource_path" is the path of ocr resources. Please refer to Developer Guide for more details.
const wchar_t* ocr_resource_path = L"D:/ocr_resources"; // Add OCR resource file path here.

if (ocr_resource_path == L"") {
    std::cout << "ocr_resource_path is still empty. Please set it with a valid path to OCR resource path." << std::endl;
    return 1;
}
```

Choose the language resource

You will need to set the language used by the OCR engine into the demo code. This is done with the **FSDK_OCREngine_SetLanguages** method and is set to "English" by default.

```
// Set languages.  
FSDK_OCREngine_SetLanguages(L"English");
```

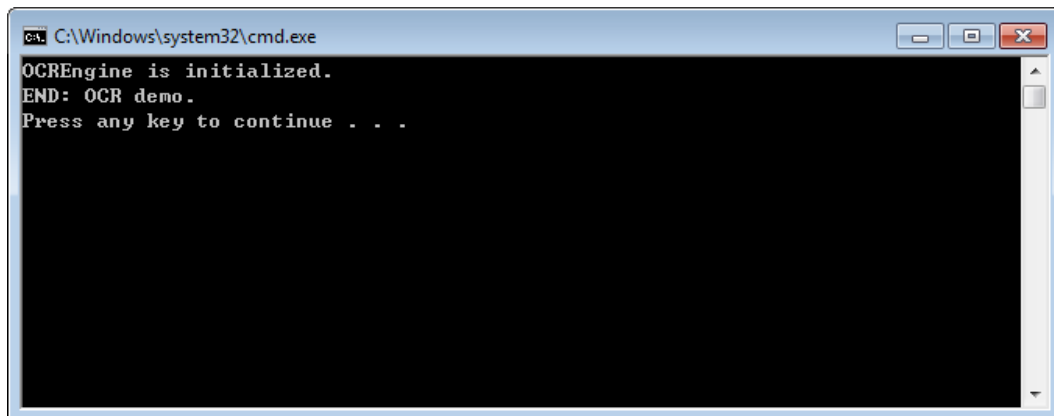
(Optional) Set log for OCREngine

If you want to print the entire log of the OCR Engine, you can call **FSDK_OCREngine_SetLogFile** method as below:

```
// Set log for OCREngine. (This can be opened to set log file if necessary)  
wchar_t output_log_file[MAX_FILE_PATH];  
swprintf_s(output_log_file, MAX_FILE_PATH, L"%socr.log", output_directory);  
FSDK_OCREngine_SetLogFile(output_log_file);
```

3.34.4.3 Run the demo

Once you run the demo successfully, the console will print the following by default:



The demo will OCR the default document ("examples\simple_demo\input_files\ocr\AboutFoxit_ocr.pdf") in four different ways, which will output four different PDFs in the output folder ("examples\simple_demo\output_files\ocr"):

- OCR Editable PDF - ocr_doc_editable.pdf
- OCR Searchable PDF - ocr_doc_searchable.pdf
- OCR Editable PDF Page - ocr_page_editable.pdf
- OCR Searchable PDF Page - ocr_page_searchable.pdf

3.35 Compliance

PDF Compliance

Foxit PDF SDK supports to convert PDF versions among PDF 1.3, PDF 1.4, PDF 1.5, PDF 1.6 and PDF 1.7. When converting to PDF 1.3, if the source document contains transparency data, then it will be converted to PDF 1.4 instead of PDF 1.3 (PDF 1.3 does not support transparency). If the source document does not contain any transparency data, then it will be converted to PDF 1.3 as expected.

PDF/A Compliance

PDF/A is an ISO-standardized version of the PDF specialized for use in the archiving and long-term preservation of electronic documents. PDF/A differs from PDF by prohibiting features unsuitable for long-term archiving, such as font linking (as opposed to font embedding), encryption, JavaScript, audio, video and so on.

Foxit PDF SDK provides APIs to convert a PDF to be compliance with PDF/A standard, or verify whether a PDF is compliance with PDF/A standard. It supports the PDF/A version including PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u (ISO 19005- 1, 19005 -2 and 19005-3).

PDF/E Compliance

PDF/E is an ISO-standardized version of the PDF specialized for the reliable exchange and archiving of engineering documents. It is designed for the creation, exchange, archiving, and printing documents used in engineering workflows.

From version 10.1, Foxit PDF SDK provides APIs to convert a PDF to be compliance with PDF/E standard or verify whether a PDF is compliance with PDF/E standard. It supports the PDF/E-1 version.

PDF/X Compliance

PDF/X is an ISO-standardized version of the PDF specialized for the exchange of graphics-intensive documents. It is mainly used to ensure consistency and predictability when printing files in fields such as design, drawing, engineering, and graphic arts.

From version 10.1, Foxit PDF SDK provides APIs to convert a PDF to be compliance with PDF/X standard or verify whether a PDF is compliance with PDF/X standard. It supports the PDF/X version including PDF/X-1a, PDF/A-3, PDF/A-4, PDF/A-4p.

Preflight Feature

From version 10.1, Foxit PDF SDK supports preflight feature, which allows users to utilize Foxit PDF SDK to get preflight keys and analyze or fixup PDF file.

This section will provide instructions on how to set up your environment for running the 'compliance' or 'preflight' demo.

3.35.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'Compliance' module permission in the license key

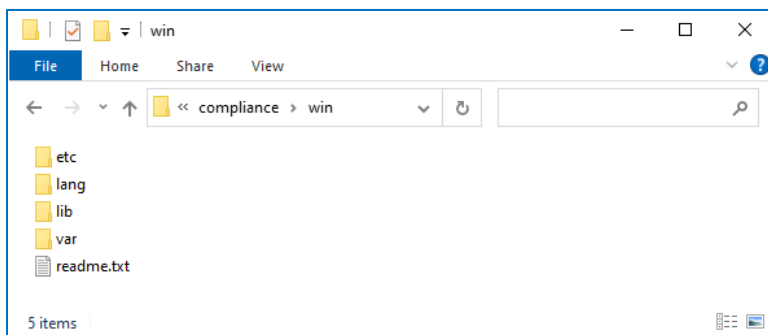
SDK Version: Foxit PDF SDK (C++, Java, C#, Objective-C) 6.4 or higher (for PDF Compliance, it requires Foxit PDF SDK 7.1 or higher); Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

Note: For PDF/E, PDF/X, and preflight feature, it requires Foxit PDF SDK 10.1.

3.35.2 Compliance resource files

Please contact Foxit support team or sales team to get the Compliance resource files package.

After getting the package, extract it to a desired directory (for example, extract the package to a directory: "**compliance/win**"), and then you can see the resource files for Compliance are as follows:



3.35.3 How to run the compliance or preflight demo

Before version 10.1, Foxit PDF SDK provides a **compliance** demo located in the "\examples\simple_demo\compliance" folder to show you how to use Foxit PDF SDK to verify whether a PDF is compliance with PDF/A standard, and convert a PDF to be compliance with PDF/A standard, as well as convert PDF versions.

From version 10.1, Foxit PDF SDK provides two demos:

- A **compliance** demo located in the "\examples\simple_demo\compliance" folder to show you how to use Foxit PDF SDK to verify whether a PDF is compliance with PDF/A or PDF/E or

PDF/X standard, and convert a PDF to be compliance with PDF/A or PDF/E or PDF/X standard, as well as convert PDF versions.

- A **preflight** demo located in the "\\examples\\simple_demo\\preflight" folder to show you how to use Foxit PDF SDK to get preflight keys and analyze or fixup PDF file.

3.35.3.1 Build a compliance resource directory

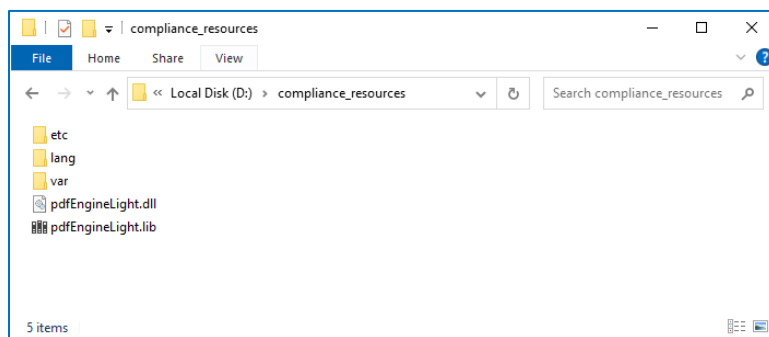
Before running the **compliance** or **preflight** demo, you should first build a compliance resource directory, and then pass the directory to Foxit PDF SDK API **FSDK_ComplianceEngine_Initialize** to initialize compliance engine.

Starting from version 10.0, the compliance resource files provide default thread-safety. For multithreading, the API **FSDK_ComplianceEngine_InitializeThreadContext** should be called first for a new thread before using any other methods in the compliance add-on module.

To build a compliance resource directory, please follow the steps below:

- 1) Create a new folder to add the resources. For example, "D:/compliance_resources".
- 2) Copy the whole folders of "**ect**", "**lang**", "**var**" under the "compliance/win" to "D:/compliance_resources".
- 3) Add the appropriate library resource based on the platform architecture.
 - For **win32**, copy **all the files** under "compliance/win/lib/x86" folder to "D:/compliance_resources".
 - For **win64**, copy **all the files** under "compliance/win/lib/x64" folder to "D:/compliance_resources".

For example, use **win32** platform architecture, then the compliance resource directory should be as follows:



3.35.3.2 Configure the demo

After building the compliance resource directory, configure the demo in the "`examples\simple_demo\compliance\compliance.c`" for compliance demo or "`examples\simple_demo\preflight\preflight.c`" for preflight demo.

Specify the compliance resource directory

In the "`compliance.c`" or "`preflight.c`" file, add the compliance resource directory as follows, which will be used to initialize the compliance engine.

```
// "compliance_resource_folder_path" is the path of compliance resource folder. Please refer to Developer Guide for more details.
const wchar_t* compliance_resource_folder_path = L"D:/compliance_resources";
// If you use an authorization key for Foxit PDF SDK, please set a valid unlock code string to compliance_engine_unlockcode for ComplianceEngine.
// If you use an trial key for Foxit PDF SDK, just keep compliance_engine_unlockcode as an empty string.
const char* compliance_engine_unlockcode = "";

FS_PDFACOMPLIANCE_HANDLE pdfa_compliance = NULL;
wchar_t save_pdf_path[MAX_FILE_PATH];
FS_PDFCOMPLIANCE_HANDLE pdf_compliance = NULL;

if (wcslen(compliance_resource_folder_path) == 0) {
    printf("compliance_resource_folder_path is still empty. Please set it with a valid path to compliance resource folder path.\n");
    return 1;
}
// Initialize compliance engine.
error_code = FSDK_ComplianceEngine_Initialize(compliance_resource_folder_path, compliance_engine_unlockcode);
```

Note:

- If you are using a trial key for Foxit PDF SDK, you do not need to authorize the compliance engine library.
- If you are using an authorization key for Foxit PDF SDK, Foxit sales team will send you an extra unlock code for initializing compliance engine library. Pass the unlock code to the **initialize** function "`FSDK_ComplianceEngine_Initialize(compliance_resource_folder_path, compliance_engine_unlockcode)`".

(Optional) Set language for compliance engine (only for compliance demo)

FSDK_ComplianceEngine_SetLanguage function is used to set language for compliance engine. The default language is "English", and the supported languages are as follows:

"Czech", "Danish", "Dutch", "English", "French", "Finnish", "German", "Italian", "Norwegian", "Polish", "Portuguese", "Spanish", "Swedish", "Chinese-Simplified", "Chinese-Traditional", "Japanese", "Korean".

For example, set the language to "Chinese-Simplified".

```
// Set languages. If not set language to ComplianceEngine, "English" will be used as default.
FSDK_ComplianceEngine_SetLanguage("Chinese-Simplified");
```

(Optional) Set a temp folder for compliance engine (only for compliance demo)

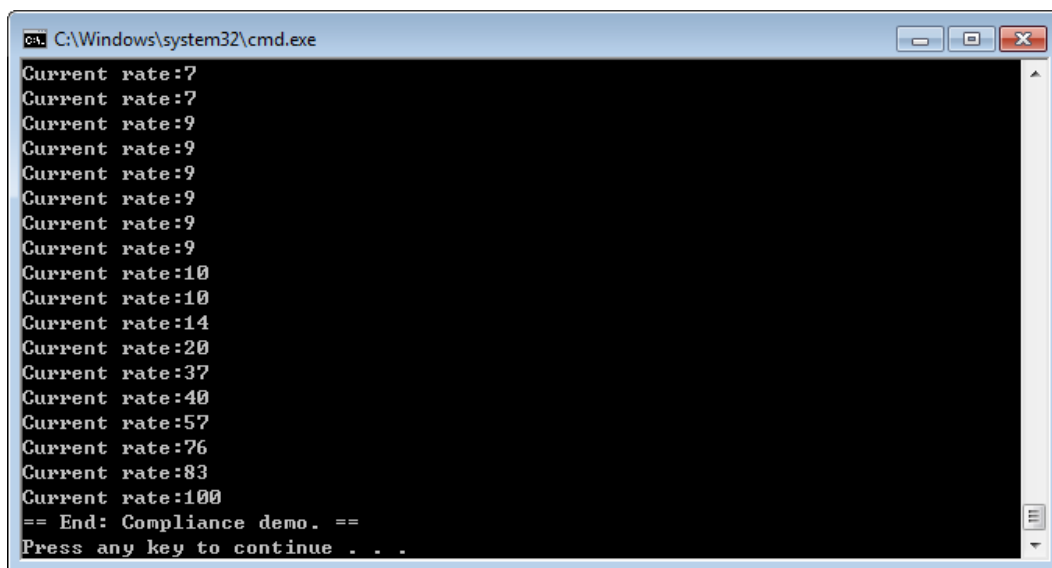
FSDK_ComplianceEngine_SetTempFolderPath function is used to set a temp folder to store several files for proper processing (e.g verifying or converting). If no custom temp folder is set by this function, the default temp folder in system will be used.

For example, set the path to "D:/compliance_temp" (should be a valid path).

```
// Set custom temp folder path for ComplianceEngine.  
FSDK_ComplianceEngine_SetTempFolderPath(L"D:/compliance_temp");
```

3.35.3.3 Run the demo

Take **compliance** demo as an example, once you run the demo successfully, the console will print the following by default:



```
ca: C:\Windows\system32\cmd.exe  
Current rate:7  
Current rate:7  
Current rate:9  
Current rate:9  
Current rate:9  
Current rate:9  
Current rate:9  
Current rate:10  
Current rate:10  
Current rate:14  
Current rate:20  
Current rate:37  
Current rate:40  
Current rate:57  
Current rate:76  
Current rate:83  
Current rate:100  
== End: Compliance demo. ==  
Press any key to continue . . .
```

The output files are located in "\\examples\\simple_demo\\output_files\\compliance" folder.

3.36 Optimization

Optimization feature can reduce the size of PDF files to save disk space and make files easier to send and store, through compressing images, deleting redundant data, discarding useless user data and so on. Optimization module provides functions to compress the color/grayscale/monochrome images in PDF files to reduce the size of the PDF files.

Note: To use the Optimization feature, please make sure the license key has the permission of the 'Optimization' module.

Example:

3.36.1 How to optimize PDF files by compressing the color/grayscale/monochrome images

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_optimization_c.h"

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file, &doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) {
    FSDK_PDFDoc_Release(doc);
    printf("Error: %d\n", error_code);
    return 1;
}
Optimization_Pause pause(0,true);
FS_OPTIMIZERSETTINGS_HANDLE settings;
FSDK_OptimizerSettings_Create(&settings);
FS_PROGRESSIVE_HANDLE progressive = NULL;
FSDK_Optimizer_Optimize(doc, settings, &pause, &progressive);
printf("Optimized Start.\n");
FSState progress_state = e_FSToBeContinued;
int percent = 0;
while (e_FSToBeContinued == progress_state) {
    FSDK_Progressive_Continue(progressive, &progress_state);
    FSDK_Progressive_GetRateOfProgress(progressive, &percent);
    char res_string[60];
    sprintf_s(res_string, 60, "Optimize progress percent: %d %", percent);
    printf("%s\n", res_string);
}
if(e_FSFinished == progress_state)
{
    wchar_t output_directory_optimized[60];
    swprintf_s(output_directory_optimized, 60, L"%lsImageCompression_Optimized.pdf", output_directory);
    FS_BOOL return_result = false;
    FSDK_PDFDoc_SaveAs(doc, output_directory_optimized, e_FSSaveFlagRemoveRedundantObjects,
    &return_result);
}
printf("Optimized Finish.\n");
```

3.37 HTML to PDF Conversion

For some large HTML files or a webpage which contain(s) many contents, it is not convenient to print or archive them directly. Foxit PDF SDK provides APIs to convert the online webpage or local HTML files like invoices or reports into PDF file(s), which makes them easier to print or archive. In the process of conversion from HTML to PDF, Foxit PDF SDK also supports to create and add PDF Tags based on the organizational structure of HTML.

For HTML to PDF module, it supports HTML5, CSS3 and JavaScript.

From version 8.1, the converted files (generated by HTML to PDF) can be provided in the form of file stream. If you want to use this feature, you should contact Foxit support team or sales team to get the latest engine files package.

This section will provide instructions on how to set up your environment for running the 'html2pdf' demo.

3.37.1 System requirements

Platform: Windows, Linux (x86 and x64), Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'Conversion' module permission in the license key

SDK Version: Foxit PDF SDK (C++, Java, C#, Objective-C) 7.0 or higher; Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.37.2 HTML to PDF engine files

Please contact Foxit support team or sales team to get the HTML to PDF engine files package.

After getting the package, extract it to a desired directory (for example, extract the package to a directory: "**htmltopdf/win/**").

3.37.3 How to run the html2pdf demo

Foxit PDF SDK provides a html2pdf demo located in the "**examples\simple_demo\html2pdf**" folder to show you how to use Foxit PDF SDK to convert from html to PDF.

3.37.3.1 Prepare a HTML2PDF engine directory

Before running the html2pdf demo, you should first extract engine package to a desired directory (for example, extract the package to a directory: "**D:/htmltopdf/win/**"), and then pass the engine file path to the API **FSDK_Convert_FromHTML** to convert html to PDF file.

3.37.3.2 Configure the demo

For html2pdf demo, you can configure the demo in the "**examples\simple_demo\html2pdf\html2pdf.c**" file, or you can configure the demo with parameters directly in a command prompt or a terminal. Following will configure the demo in "html2pdf.c" file.

Specify the html2pdf engine directory

In the "html2pdf.c" file, add the path of the engine file "fxhtml2pdf.exe" as follows, which will be used to convert html files to PDF files.

```
// "engine_path" is the path of the engine file "fxhtml2pdf" which is used to convert html to pdf. Please refer to Developer Guide for more details.  
const wchar_t* engine_path = L"D:/htmltopdf/win/fxhtml2pdf.exe"; // or engine_path = L"D:/htmltopdf/win/fxhtml2pdf"
```

(Optional) Specify cookies file path

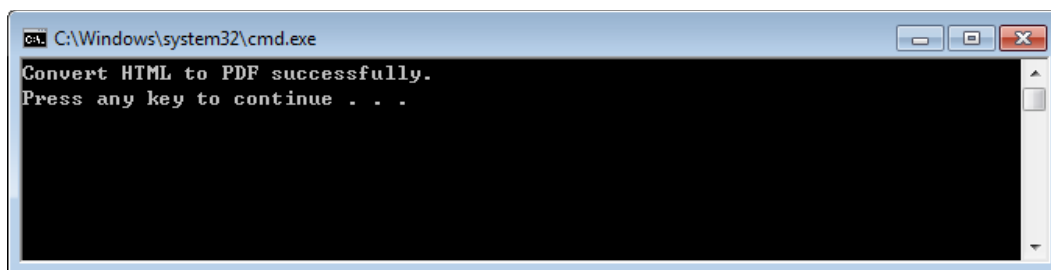
Add the path of the cookies file exported from the web pages that you want to convert. For example,

```
// "cookies_path" is the path of the cookies file exported from the web pages that you want to convert. Please refer to Developer Guide for more details.  
const wchar_t* cookies_path = L"D:/cookies.txt";
```

3.37.3.3 Run the demo

Run the demo without parameters

Once you run the demo successfully, the console will print the following by default:

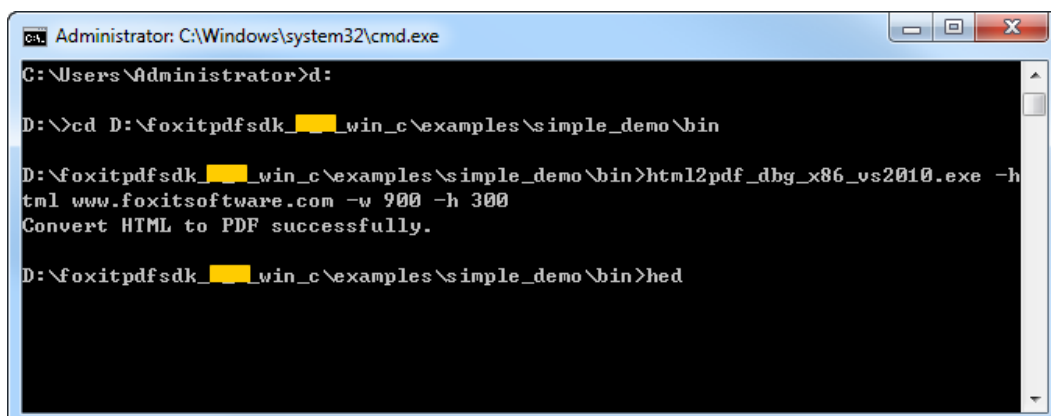


```
C:\Windows\system32\cmd.exe  
Convert HTML to PDF successfully.  
Press any key to continue . . .
```

Run the demo with parameters

After building the demo successfully, open a command prompt, navigate to "examples\simple_demo\bin", type "html2pdf_dbg_x86_vs2010.exe --help" for example to see how to use the parameters to execute the program.

For example, convert the URL web page "www.foxitsoftware.com" into a PDF with setting the page width to 900 points and the page height to 300 points:



```
Administrator: C:\Windows\system32\cmd.exe  
C:\Users\Administrator>d:  
D:\>cd D:\foxitpdfsdk\win_c\examples\simple_demo\bin  
D:\foxitpdfsdk\win_c\examples\simple_demo\bin>html2pdf_dbg_x86_vs2010.exe -h  
tml www.foxitsoftware.com -w 900 -h 300  
Convert HTML to PDF successfully.  
D:\foxitpdfsdk\win_c\examples\simple_demo\bin>hed
```

The output file is located in "\\examples\\simple_demo\\output_files\\html2pdf" folder.

Parameters Description

Basic Syntax:

```
html2pdf_xxx <-html <The url or html path>> <-o <output pdf path>> <-engine <htmltopdf engine path>>
    [-w <page width>] [-h <page height>] [-ml <margin left>] [-mr <margin right>]
    [-mt <margin top>] [-mb <margin bottom>] [-r <page rotation degree>] [-mode <page
    mode>] [-scale <scaling mode>] [-link <whether to convert link>]
    [-tag <whether to generate tag>] [-bookmarks <whether to generate bookmarks>]
    [-print_background <whether to print background>]
    [-optimize_tag <whether to optimize tag tree>] [-media <media style>] [-encoding <HTML
    encoding format>] [-render_images <Whether to render images>]
    [-remove_underline_for_link <Whether to remove underline for link>]
    [-headerfooter <Whether to generate headerfooter>] [-headerfooter_title <headerfooter
    title>] [-headerfooter_url <headerfooter url>] [-bookmark_root_name <bookmark root
    name>] [-resize_objects <Whether to enable the JavaScripts related resizing of the objects>]
    [-cookies <cookies file path>] [-timeout <timeout>] [--help<Parameter usage>]
```

Note:

- <> required
- [] optional

Parameters	Description
--help	The usage description of the parameters.
-html	The url or html file path. For examples '-html www.foxitsoftware.com'.
-o	The path of the output PDF file.
-engine	The path of the engine file "fxhtml2pdf.exe".
-w	The page width of the output PDF file in points.
-h	The page height of the output PDF file in points.
-r	The page rotation for the output PDF file. <ul style="list-style-type: none"> • 0 : 0 degree. • 1 : 90 degree. • 2 : 180 degree. • 3 : 270 degree.
-ml	The left margin of the pages for the output PDF file.
-mr	The right margin of the pages for the output PDF file.

Parameters	Description
-mt	The top margin of the pages for the output PDF file.
-mb	The bottom margin of the pages for the output PDF file.
-mode	The page mode for the output PDF file. <ul style="list-style-type: none">• 0 : Single page mode.• 1 : Multiple pages mode.
-scale	The scaling mode. <ul style="list-style-type: none">• 0 : No need to scale pages.• 1 : Scale pages.• 2 : Enlarge page.
-link	Whether to convert links. <ul style="list-style-type: none">• 'yes' : Convert links.• 'no' : No need to convert links.
-tag	Whether to generate tag. <ul style="list-style-type: none">• 'yes' : Generate tag.• 'no' : No need to generate tag.
-bookmarks	Whether to generate bookmarks. <ul style="list-style-type: none">• 'yes' : Generate bookmarks .• 'no' : No need to generate bookmarks.
-print_background	Whether to print background. <ul style="list-style-type: none">• 'yes' : Print bookmarks .• 'no' : No need to print bookmarks.
-optimize_tag	Whether to optimize tag tree. <ul style="list-style-type: none">• 'yes' : Optimize tag tree .• 'no' : No need to optimize tag tree.
-media	The media style. <ul style="list-style-type: none">• 0 : Screen media style.• 1 : Print media style.
-encoding	The HTML encoding format. <ul style="list-style-type: none">• 0 : Auto encoding .• 1-73 : Other encodings.
-render_images	Whether to render images. <ul style="list-style-type: none">• 'yes' : Render images.• 'no' : No need to render images.
-remove_underline_for_link	Whether to remove underline for link.

Parameters	Description
	<ul style="list-style-type: none"> 'yes' : Remove underline for link. 'no' : No need to remove underline for link.
-headerfooter	Whether to generate headerfooter. <ul style="list-style-type: none"> 'yes' : Generate headerfooter. 'no' : No need to generate headerfooter.
-headerfooter_title	The headerfooter title.
-headerfooter_url	The headerfooter url.
-bookmark_root_name	The bookmark root name.
-resize_objects	Whether to enable the JavaScripts related resizing of the objects during rendering process. <ul style="list-style-type: none"> 'yes' : Enable. 'no' : Disable.
-cookies	The path of the cookies file exported from a URL that you want to convert.
-timeout	The timeout of loading webpages.

3.37.4 How to work with Html2PDF API

```
#include "include/fs_basictypes_c.h"
#include "include/fs_convert_c.h"

FSHTML2PDFSettingData pdf_setting_data;
pdf_setting_data.is_convert_link = true;
pdf_setting_data.is_generate_tag = true;
pdf_setting_data.to_generate_bookmarks = true;
pdf_setting_data.rotate_degrees = 0;
pdf_setting_data.page_height = 640;
pdf_setting_data.page_width = 900;
pdf_setting_data.page_mode = e_FSPageModeSinglePage;
pdf_setting_data.scaling_mode = e_FSScalingModeScale;
pdf_setting_data.to_print_background = true;
pdf_setting_data.to_optimize_tag_tree = false;
pdf_setting_data.media_style = e_FSMediaStyleScreen;
...

FSDK_Convert_FromHTML(url_or_html, engine_path, cookies_path, pdf_setting_data, output_path, time_out);
```

3.37.5 How to get HTML data from stream and convert it to a PDF file

1. Defines two [FileRead](#) structure variables named "[filereadercallback](#)" and "[filereadercallbackpng](#)" used to get HTML data from a stream or memory. Additionally, defines a [FileWriter](#) structure variable named "[filewritercallback](#)" used to do file writing. For the initializing of the

"filereadercallback", "filereadercallbackpng" and "filewritercallback" variables, please refer to the **html2pdf** demo located in the "\examples\simple_demo\html2pdf" folder.

2. Get html data from stream and set resources related to source html.
3. Call the [FSDK_Convert_FromHTML2](#) function to convert it to a PDF file.

```
# include "include/fs_basictypes_c.h"
# include "include/fs_common_c.h"
# include "include/fs_pdfdoc_c.h"
# include "include/fs_convert_c.h"

....
// Get HTML data from stream
if (isfilestreamload)
{
    FSHTML2PDFSettingData pdf_setting_data;
    pdf_setting_data.page_height = 650;
    pdf_setting_data.page_width = 950;
    pdf_setting_data.is_to_page_scale = false;
    FSRectF rectf;
    rectf.left = 18;
    rectf.bottom = 18;
    rectf.right = 18;
    rectf.top = 18;
    pdf_setting_data.page_margin = rectf;
    pdf_setting_data.is_convert_link = true;
    pdf_setting_data.rotate_degrees = e_FSRotation0;
    pdf_setting_data.is_generate_tag = true;
    pdf_setting_data.page_mode = e_FSPageModeSinglePage;
    pdf_setting_data.to_generate_bookmarks = true;
    pdf_setting_data.scaling_mode = e_FSScalingModeNone;
    pdf_setting_data.encoding_format = e_FSEncodingFormatDefault;
    pdf_setting_data.to_render_images = true;
    pdf_setting_data.to_remove_underline_for_link = false;
    pdf_setting_data.to_set_headerfooter = false;
    pdf_setting_data.headerfooter_title.str = L"";
    pdf_setting_data.headerfooter_title.len = 0;
    pdf_setting_data.headerfooter_url.str = L"";
    pdf_setting_data.bookmark_root_name.str = L"abcde";
    pdf_setting_data.bookmark_root_name.len = wcslen(L"abcde");
    pdf_setting_data.to_resize_objects = true;
    pdf_setting_data.to_print_background = false;
    pdf_setting_data.to_optimize_tag_tree = false;
    pdf_setting_data.media_style = e_FSMediaStyleScreen;
    pdf_setting_data.to_load_active_content = false;
    const wchar_t* cookies = L"";
    FILE* filepng = NULL;
    FILE* file = NULL;
    FILE* file_writer = NULL;
    wchar_t output_file[MAX_FILE_PATH];
```

```

swprintf_s(output_file, MAX_FILE_PATH, L"%lshtml2pdf_filestream_result.pdf", output_directory);
_wfopen_s(&file_writer, output_file, L"w+b");
filewritercallback->user_data = filewritercallback;
filewritercallback->Flush = gFlushWriter;
filewritercallback->GetSize = gGetSizeWriter;
filewritercallback->WriteBlock = gWriteBlockWriter;
filewritercallback->Release = gReleaseWriter;
// "htmlfile" is the path of the html file to be loaded. For example: "C:/aaa.html". The method of "FromHTML"
will load this file as a stream.
const wchar_t* htmlfile = L"";
_wfopen_s(&file, htmlfile, L"rb");
filereadercallback->user_data = filereadercallback;
filereadercallback->GetSize = gGetSize;
filereadercallback->ReadBlock = gReadBlock;
filereadercallback->Release = gRelease;
FSHTML2PDFRelatedResource html2PDFRelatedResource[1];

// "htmlfilepng" is the path of the png resource file to be loaded. For example: "C:/aaa.png". set it in the
related_resource_file of HTML2PDFRelatedResource.
const wchar_t* htmlfilepng = L"";
filereadercallbackpng->user_data = filereadercallbackpng;
filereadercallbackpng->GetSize = gGetSizePng;
filereadercallbackpng->ReadBlock = gReadBlockPng;
filereadercallbackpng->Release = gReleasePng;
_wfopen_s(&filepng, htmlfilepng, L"rb");
html2PDFRelatedResource[0].related_resource_file = (FSReaderCallback*)filereader1;

// "relativefilepath" is the resource file's relative path. For example: "./aaa.png".
const char_t* relativefilepath = "";
size_t convertedChars = 0;
const char_t* curLocale = setlocale(LC_ALL, NULL);
setlocale(LC_ALL, "chs");
const char* source = relativefilepath;
char_num = sizeof(char)* strlen(relativefilepath) + 1;
desthtmlfilepng = malloc(sizeof(wchar_t) * char_num);
mbstowcs_s(&converted_chars, desthtmlfilepng, char_num, source, _TRUNCATE);

html2PDFRelatedResource[0].resource_file_relative_path.str = desthtmlfilepng;
html2PDFRelatedResource[0].resource_file_relative_path.len = wcslen(desthtmlfilepng);
FSErrorCode error_code = FSDK_Convert_FromHTML2(filereader, html2PDFRelatedResource, 1, engine_path,
NULL, pdf_setting_data, filewrite, 15);
free(desthtmlfilepng);

```

3.38 Office to PDF Conversion with third-party engines

From version 7.3, Foxit PDF SDK provides APIs to convert Microsoft Office documents (Word and Excel) into professional-quality PDF files on Windows platform.

From version 7.4, Foxit PDF SDK also supports to convert PowerPoint documents into PDF files on Windows platform.

From version 8.4, Foxit PDF SDK provides APIs to convert Microsoft Office documents (Word, Excel and PowerPoint) into professional-quality PDF files on Linux platform (x86, x64 and armv8). Foxit PDF SDK for C API just supports Windows platform.

For using this feature, please note that:

- Make sure that Microsoft Office 2007 version or higher is already installed on your Windows system.
- Before converting Excel to PDF, make sure that the default Microsoft virtual printer is already set on your Windows system.

3.38.1 System requirements

Platform: Windows, Linux (x86, x64 and armv8)

Programming Language: C, C++, Python, Java, C#, Node.js

License Key requirement: 'Conversion' module permission in the license key

SDK Version: Word and Excel (Foxit PDF SDK (C++, C#, Java) 7.3 or higher; Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher); PowerPoint (Foxit PDF SDK (C, C++, C#, Java) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher); Word/Excel/PowerPoint (Foxit PDF SDK (Node.js) 10.0 or higher)

Example:

3.38.2 How to convert Word to PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_convert_c.h"
...
// Make sure that SDK has already been initialized successfully.

const wchar_t* word_file_path = L"test.doc";
const wchar_t* output_path = L"saved.pdf";

// Use default Word2PDFSettingData values.
FSWord2PDFSettingData word_convert_setting_data;
word_convert_setting_data.include_doc_props = false;
word_convert_setting_data.optimize_option = e_FSConvertOptimizeOptionForPrint;
word_convert_setting_data.content_option = e_FSConvertContentOptionOnlyContent;
word_convert_setting_data.bookmark_option = e_FSConvertBookmarkOptionNone;
word_convert_setting_data.convert_to_pdfa = false;
FSDK_Convert_FromWord(word_file_path, L"", output_path, word_convert_setting_data);
```

3.38.3 How to convert Excel to PDF

```
#include "include/fs_basictypes_c.h"
```

```
#include "include/fs_convert_c.h"
...
// Make sure that SDK has already been initialized successfully.

const wchar_t* excel_file_path = L"test.xls";
const wchar_t* output_path = L"saved.pdf";

// Use default Excel2PDFSettingData values.
FSExcel2PDFSettingData excel_convert_setting_data;
excel_convert_setting_data.include_doc_props = false;
excel_convert_setting_data.quality = e_FSConvertQualityStandard;
excel_convert_setting_data.ignore_print_area = true;
excel_convert_setting_data.scale_type = e_FSScaleTypeNone;
excel_convert_setting_data.convert_to_pdfa = false;
FSDK_Convert_FromExcel(excel_file_path, L"", output_path, excel_convert_setting_data);
```

3.38.4 How to convert PowerPoint to PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_convert_c.h"
...
// Make sure that SDK has already been initialized successfully.

const wchar_t* ppt_file_path = L"test.ppt";
const wchar_t* output_path = L"saved.pdf";

// Use default PowerPoint2PDFSettingData values.
FSPowerPoint2PDFSettingData ppt_convert_setting_data;
ppt_convert_setting_data.intent = e_FSConvertIntentPrint;
ppt_convert_setting_data.frame_output_slides = false;
ppt_convert_setting_data.output_type = e_FSOutputSlides;
ppt_convert_setting_data.handout_order = e_FSHandoutOrderVerticalFirst;
ppt_convert_setting_data.output_hidden_slides = false;
ppt_convert_setting_data.include_doc_props = false;
FSDK_Convert_FromPowerPoint(ppt_file_path, L"", output_path, ppt_convert_setting_data);
```

3.39 Office to PDF Conversion without third-party engines

From version 10.1, Foxit PDF SDK offers the capability to convert Microsoft Office documents (Word, Excel and PowerPoint) into professional-quality PDF files without any third-party engine. This feature is available through the Foxit PDF Conversion SDK on Windows platform.

3.39.1 System requirements

Platform: Windows

Programming Language: C, C++, Python, Java, C#, Node.js

License Key requirement: 'Office2PDF' module permission in the license key

SDK Version: Foxit PDF SDK 10.1

3.39.2 Office to PDF resource files (Foxit PDF Conversion SDK)

Please contact Foxit support team or sales team to get the Foxit PDF Conversion SDK (C++).

After getting Foxit PDF Conversion SDK package, extract it to a desired directory, for example, extract the package to a directory: "D:/foxitpdfconversionsdk*_win/" for Windows.

3.39.3 How to run the office2pdf demo using Foxit PDF Conversion SDK

Before running the office2pdf demo in the "\examples\simple_demo\office2pdf" folder using Foxit PDF Conversion SDK, you should first add the Foxit PDF Conversion SDK library in the demo code, for example:

```
// If you want to convert office files to PDF whitout other third-party engines, you can use the Office2PDF module.  
const wchar_t* library_path = L"D:/foxitpdfconversionsdk*_win/lib/fpdfconversionsdk_win32.dll";
```

Then, specify the office2pdf resource data files:

```
// A valid path of a folder which contains resource data files.  
office2pdf_setting_data.resource_folder_path.str = L"D:/foxitpdfconversionsdk*_win/res/office2pdf";
```

Finally, run the demo following the steps as the other demos.

3.39.4 How to convert office files to PDF without third-party engines

```
#include "include/fs_basictypes_c.h"  
#include "include/fs_common_c.h"  
#include "include/fs_office2pdf_c.h"  
...  
  
// If you want to convert office files to PDF whitout other third-party engines, you can use the Office2PDF module.  
const wchar_t* library_path = L""; // Path of Foxit PDF Conversion SDK library, please ensure the path is valid.  
// Initialize the Office2PDF module.  
FSDK_Office2PDF_Initialize(library_path);  
FSDK_Office2PDF_SettingData office2pdf_setting_data;  
// A valid path of a folder which contains resource data files.  
office2pdf_setting_data.resource_folder_path.str = L"";  
office2pdf_setting_data.resource_folder_path.len = 0;  
office2pdf_setting_data.is_embed_font = FALSE;  
office2pdf_setting_data.word_setting_data.is_generate_bookmark = FALSE;  
office2pdf_setting_data.excel_setting_data.is_separate_workbook = FALSE;  
office2pdf_setting_data.excel_setting_data.is_output_hidden_workbook = FALSE;  
office2pdf_setting_data.excel_setting_data.workbook_names_array = NULL;  
office2pdf_setting_data.excel_setting_data.workbook_names_array_length = 0;  
  
// Conver Word file to PDF file.  
swprintf_s(output_file, MAX_FILE_PATH, L"%lsword2pdf_result_foxit.pdf", output_directory);
```

```
error_code = FSDK_Office2PDF_ConvertFromWord(word_file_path, L"", output_file, office2pdf_setting_data,
&ret_value);
// Conver Excel file to PDF file.
swprintf_s(output_file, MAX_FILE_PATH, L"%lsexcel2pdf_result_foxit.pdf", output_directory);
error_code = FSDK_Office2PDF_ConvertFromExcel(excel_file_path, L"", output_file, office2pdf_setting_data,
&ret_value);
// Conver PowerPoint file to PDF file.
swprintf_s(output_file, MAX_FILE_PATH, L"%lsppt2pdf_result_foxit.pdf", output_directory);
error_code = FSDK_Office2PDF_ConvertFromPowerPoint(ppt_file_path, L"", output_file, office2pdf_setting_data,
&ret_value);
// Release the Office2PDF module.
FSDK_Office2PDF_Release();
```

3.40 Output Preview

Foxit PDF SDK supports output preview feature which can preview color separations and test different color profiles.

3.40.1 System requirements

Platform: Windows, Linux (x86 and x64), Mac (x64)

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: valid license key

SDK Version: Foxit PDF SDK (C, C++, Java, C#, Objective-C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.40.2 How to run the output preview demo

Before running the output preview demo in the "\\examples\\simple_demo\\output_preview" folder, you should first set the folder path of "\\res\\icc_profile" in the SDK package to the variable **default_icc_folder_path**. For example:

```
// "default_icc_folder_path" is the path of the folder which contains default icc profile files. Please refer to
Developer Guide for more details.
const wchar_t* default_icc_folder_path = L"E:/foxitpdfsdk_X_X_win_c/res/icc_profile";
```

Then, run the demo following the steps as the other demos.

3.40.3 How to do output preview using Foxit PDF SDK

```
#include "include/fs_common_c.h"
#include "include/fs_outputpreview_c.h"

// Make sure that SDK has already been initialized successfully.

// Set folder path which contains default icc profile files.
FSDK_ErrorCode error_code = FSDK_Library_SetDefaultICCProfilesPath(default_icc_folder_path);
```

```
// Load a PDF document; Get a PDF page and parse it.
// Prepare a Renderer object and the matrix for rendering.

FS_OUTPUTPREVIEW_HANDLE output_preview;
FSDK_OutputPreview_Create(pdf_doc, &output_preview);
wchar_t simulation_icc_file_path[MAX_FILE_PATH] = { 0 };
swprintf_s(simulation_icc_file_path, MAX_FILE_PATH, L"%lsicc_profile/USWebCoatedSWOP.icc", input_path);
FSDK_OutputPreview_SetSimulationProfile(output_preview, simulation_icc_file_path);
FSDK_OutputPreview_SetShowType(output_preview, e_FSShowAll);

FS_UINT32 process_plates_length = 0;
FSDK_OutputPreview_GetPlates(output_preview, e_FSColorantTypeProcess, NULL, &process_plates_length);
process_plates = (FS_BSTR*)malloc(sizeof(FS_BSTR) * process_plates_length);
FSDK_OutputPreview_GetPlates(output_preview, e_FSColorantTypeProcess, process_plates,
process_plates_length);

FS_UINT32 spot_plates_length = 0;
FSDK_OutputPreview_GetPlates(output_preview, e_FSColorantTypeSpot, NULL, &spot_plates_length);
spot_plates = (FS_BSTR*)malloc(sizeof(FS_BSTR) * spot_plates_length);
FSDK_OutputPreview_GetPlates(output_preview, e_FSColorantTypeSpot, spot_plates, &spot_plates_length);

// Set check status of process plate to be true, if there's any process plate.
for (int i = 0; i < (int)process_plates_length; i++) {
    error_code = FSDK_OutputPreview_SetCheckStatus(output_preview, process_plates[i], true);
}
free(process_plates);

// Set check status of spot plate to be true, if there's any spot plate.
for (int i = 0; i < (int)spot_plates_length; i++) {
    error_code = FSDK_OutputPreview_SetCheckStatus(output_preview, spot_plates[i], true);
}
free(spot_plates);

// Generate preview bitmap
FS_BITMAP_HANDLE preview_bitmap;
error_code = FSDK_OutputPreview_GeneratePreviewBitmap(output_preview, pdf_page, display_matrix,
renderer, &preview_bitmap);
```

3.41 Combination

Combination feature is used to combine several PDF files into one PDF file.

3.41.1 How to combine several PDF files into one PDF file

```
#include "include/fs_common_c.h"
#include "include/fs_combination_c.h"

// Make sure that SDK has already been initialized successfully.
```

```
int info_array_size = 3;
FS_COMBINEDOCUMENTINFO_HANDLE* info_array = malloc(sizeof(FS_COMBINEDOCUMENTINFO_HANDLE) *
info_array_size);
FS_WSTR password;
password.str = NULL;
password.len = 0;
auto item = info_array[0];
wchar_t input_file[MAX_FILE_PATH] = { 0 };
swprintf_s(input_file, MAX_FILE_PATH, L"%lsAboutFoxit.pdf", input_path);
FSDK_CombineDocumentInfo_Create((input_path), &password, &info_array[0]);
item = info_array[0];
swprintf_s(input_file, MAX_FILE_PATH, L"%lsAnnot_all.pdf", input_path);
FSDK_CombineDocumentInfo_Create(input, &password, &info_array[1]);
item = info_array[1];
swprintf_s(input_file, MAX_FILE_PATH, L"%lsSamplePDF.pdf", input_path);
FSDK_CombineDocumentInfo_Create(input_file, &password, info_array[2]);
item = info_array[2];

FS_UINT32 option = e_FSCombineDocsOptionBookmark | e_FSCombineDocsOptionAcroformRename
| e_FSCombineDocsOptionStructueTree | e_FSCombineDocsOptionOutputIntents
| e_FSCombineDocsOptionOCProperties | e_FSCombineDocsOptionMarkInfos
| e_FSCombineDocsOptionPageLabels | e_FSCombineDocsOptionNames
| e_FSCombineDocsOptionObjectStream | e_FSCombineDocsOptionDuplicateStream;

FS_PROGRESSIVE_HANDLE progressive = NULL;
wchar_t output_file[MAX_FILE_PATH] = { 0 };
swprintf_s(output_file, MAX_FILE_PATH, L"%ls Test_Combined.pdf ", output_directory);
FSDK_Combination_StartCombineDocuments(output_file, info_array, info_array_size, option, NULL,
&progressive);
FSState progress_state = e_FSToBeContinued;
while (e_FSToBeContinued == progress_state) {
    FSDK_Progressive_Continue(progressive, &progress_state);
}
for (int i = 0; i < info_array_size; i++) {
    FSDK_CombineDocumentInfo_Release(info_array[i]);
}
free(info_array);
```

3.42 PDF Portfolio

PDF portfolios are a combination of files with different formats. Portfolio file itself is a PDF document, and files with different formats can be embedded into this kind of PDF document.

3.42.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: valid license key

SDK Version: Foxit PDF SDK (C, C++, Java, C#, Objective-C) 7.6 or higher; Foxit PDF SDK (Python) 8.3 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

Example:

3.42.2 How to create a new and blank PDF portfolio

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_portfolio_c.h"

// Make sure that SDK has already been initialized successfully.

FS_PORTFOLIO_HANDLE new_portfolio = NULL;
FSDK_Portfolio_CreatePortfolio(&new_portfolio);

// Set properties, add file/folder node to the new portfolio.
...

// Get portfolio PDF document object.
FS_PORTFOLIONODE_HANDLE root_node = NULL;
FSDK_Portfolio_GetRootNode(new_portfolio, &root_node);
...

// Release handles when no need to use them any more.
if (root_node) {
    FSDK_PortfolioNode_Release(root_node);
    root_node = NULL;
}
if (new_portfolio) {
    FSDK_Portfolio_Release(new_portfolio);
    new_portfolio = NULL;
}
```

3.42.3 How to create a Portfolio object from a PDF portfolio

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_portfolio_c.h"

// Make sure that SDK has already been initialized successfully.

FS_PDFDOC_HANDLE portfolio_pdf_doc = NULL;
FSDK_PDFDoc_Create0(portfolio_file_path, &portfolio_pdf_doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(portfolio_pdf_doc, NULL);
if (e_FSErrSuccess == error_code) {
    FS_BOOL is_portfolio = FALSE;
    FSDK_PDFDoc_IsPortfolio(pdf_doc, &is_portfolio);
    if (is_portfolio) {
        FS_PORTFOLIO_HANDLE existed_portfolio = NULL;
```

```
FSDK_Portfolio_CreatePortfolio0(pdf_doc, &existed_portfolio);
...
// Release handles when no need to use them any more.
if (existed_portfolio) {
    FSDK_Portfolio_Release(existed_portfolio);
    existed_portfolio = NULL;
}
}
}
...

// Release handles when no need to use them any more.
if (portfolio_pdf_doc) {
    FSDK_PDFDoc_Release(portfolio_pdf_doc);
    portfolio_pdf_doc = NULL;
}
```

3.42.4 How to get portfolio nodes

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_filespec_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_portfolio_c.h"

// Make sure that SDK has already been initialized successfully.

// Portfolio object has been created, assume it is named "portfolio".
FS_PORTFOLIONODE_HANDLE root_node = NULL;
FSDK_Portfolio_GetRootNode(portfolio, &root_node);
FS_PORTFOLIOFOLDERNODE_HANDLE root_folder = NULL;
FSDK_PortfolioFolderNode_Create(root_node, &root_folder);

FS_PORTFOLIONODE_HANDLE* sub_node_array = NULL;
FS_UINT32 array_length = 0;
FSDK_PortfolioFolderNode_GetSortedSubNodes(root_folder, sub_node_array, array_length);
sub_node_array = malloc(sizeof(FS_PORTFOLIONODE_HANDLE) * array_length);
FSDK_PortfolioFolderNode_GetSortedSubNodes(root_folder, sub_node_array, array_length);
for (FS_UINT32 index = 0; index < array_length; index++) {
    FS_PORTFOLIONODE_HANDLE node = sub_node_array[index];
    FSPortfolioNodeType node_type = e_FSPortfolioNodeTypeUnknown;
    FSDK_PortfolioNode_GetNodeType(node, &node_type);
    switch (node_type) {
    case e_FSTypeFolder: {
        FS_PORTFOLIOFOLDERNODE_HANDLE folder_node = NULL;
        FSDK_PortfolioFolderNode_Create(node, &folder_node);

        // Use PortfolioFolderNode's getting method to get some properties.
        ...

        FS_PORTFOLIONODE_HANDLE* sub_node_array_2 = NULL;
        FS_UINT32 array_length_2 = 0;
```

```
FSDK_PortfolioFolderNode_GetSortedSubNodes(root_folder, sub_node_array_2, &array_length_2);
sub_node_array_2 = malloc(sizeof(FS_PORTFOLIONODE_HANDLE) * array_length_2);
FSDK_PortfolioFolderNode_GetSortedSubNodes(root_folder, sub_node_array_2, &array_length_2);
...
// Release handle
for (FS_UINT32 i = 0; i < array_length_2; i++) {
    FSDK_PortfolioNode_Release(sub_node_array_2[i]);
}
free(sub_node_array_2);
if (folder_node) {
    FSDK_PortfolioFolderNode_Release(folder_node);
    folder_node = NULL;
}
break;
}
case e_FSTypeFile: {
    FS_PORTFOLIOFILENODE_HANDLE file_node = NULL;
    FSDK_PortfolioFileNode_Create(node, &file_node);
    // Get file specification from this file node, and then get/set information from/to this file specification
    object.
    FS_FILESPEC_HANDLE file_spec = NULL;
    FSDK_PortfolioFileNode_GetFileSpec(file_node, &file_spec);
    ...
    // Release handle.
    if (file_spec) {
        FSDK_FileSpec_Release(file_spec);
        file_spec = NULL;
    }
    if (file_node) {
        FSDK_PortfolioFileNode_Release(file_node);
        file_node = NULL;
    }
    break;
}
}
}
...
// Release handles when no need to use them any more.
for (FS_UINT32 i = 0; i < array_length; i++) {
    FSDK_PortfolioNode_Release(sub_node_array[i]);
}
free(sub_node_array);
if (root_folder) {
    FSDK_PortfolioFolderNode_Release(root_folder);
    root_folder = NULL;
}
if (root_node) {
    FSDK_PortfolioNode_Release(root_node);
    root_node = NULL;
}
```

3.42.5 How to add file node or folder node

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fx_stream_c.h"
#include "include/fs_portfolio_c.h"

// Make sure that SDK has already been initialized successfully.

// Add file from path.
const wchar_t* path_to_a_file = L"directory/Sample.txt";
FS_PORTFOLIOFILENODE_HANDLE new_file_node_1 = NULL;
FSDK_PortfolioFolderNode_AddFile(root_folder, path_to_a_file, &new_file_node_1);

// User can update properties of file specification for new_file_node_1 if necessary.
...

// Add file from MyStreamCallback which is inherited from FSStreamCallback and implemented by user.
FILE* filestream = NULL;
callback_filestream_word = (FSStreamCallback*)malloc(sizeof(FSStreamCallback));
_wfopen_s(&filestream, saved_word_file_path_stream, L"wb");
my_stream_callback->Flush = gFlushFileStream;
my_stream_callback->GetPosition = gGetPositionFileStream;
my_stream_callback->GetSize = gGetSizeFileStream;
my_stream_callback->IsEOF = gIsEOFFileStream;
my_stream_callback->WriteBlock = gWriteBlockFileStream;
my_stream_callback->ReadBlock = gReadBlockFileStream;
my_stream_callback->ReadBlock0 = gReadBlock0FileStream;
my_stream_callback->Release = gReleaseFileStream;
my_stream_callback->Retain = gRetainFileStream;
my_stream_callback->user_data = callback_filestream_word;
FS_PORTFOLIOFILENODE_HANDLE new_file_node_2 = NULL;
FSDK_PortfolioFolderNode_AddFile0(root_folder, my_stream_callback, L"file_name", &new_file_node_2);

// Please get file specification of new_file_node_2 and update properties of the file specification by its setting
methods.
...

// Add a loaded PDF file.
// Open and load a PDF file, assume it is named "test_pdf_doc".
...
FS_PORTFOLIOFILENODE_HANDLE new_file_node_3 = NULL;
FSDK_PortfolioFolderNode_AddPDFDoc(root_folder, test_pdf_doc, L"pdf_file_name", &new_file_node_3);

// User can update properties of file specification for new_file_node_3 if necessary.
...

// Add a sub folder in root_folder.
FS_PORTFOLIOFOLDERNODE_HANDLE new_sub_foldernode = NULL;
FSDK_PortfolioFolderNode_AddSubFolder(root_folder, L"Sub Folder-1", &new_sub_foldernode);
```



```
// User can add file or folder node to new_sub_foldernode.
...

// Release handles when no need to use them any more.
if (new_sub_foldernode) {
    FSDK_PortfolioFolderNode_Release(new_sub_foldernode);
    new_sub_foldernode = NULL;
}
if (new_sub_filenode_3) {
    FSDK_PortfolioFileNode_Release(new_sub_filenode_3);
    new_sub_filenode_3 = NULL;
}
if (new_sub_filenode_2) {
    FSDK_PortfolioFileNode_Release(new_sub_filenode_2);
    new_sub_filenode_2 = NULL;
}
if (new_sub_filenode_1) {
    FSDK_PortfolioFileNode_Release(new_sub_filenode_1);
    new_sub_filenode_1 = NULL;
}
```

3.42.6 How to remove a node

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_portfolio_c.h"

// Make sure that SDK has already been initialized successfully.

// Remove a child folder node from its parent folder node.
FSDK_PortfolioFolderNode_RemoveSubNode(parent_folder_node, child_folder_node);
// Remove a child file node from its parent folder node.
FSDK_PortfolioFolderNode_RemoveSubNode(parent_folder_node, child_file_node);
```

3.43 Table Maker

From version 8.4, Foxit PDF SDK supports to add table to PDF files.

3.43.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'TableMaker' module permission in the license key

SDK Version: Foxit PDF SDK (C, C++, C#, Java, Python, Objective-C) 8.4 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.43.2 How to add table to a PDF document

Foxit PDF SDK provides an electronic table demo located in the "`\examples\simple_demo\electronictable`" folder to show you how to use Foxit PDF SDK to add table to PDF document.

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_tablegenerator_c.h"

// Add a spreadsheet with 4 rows and 3 columns
int index = 0;
wchar_t cell_text[12];
FSTableCellData cell_array[4][3];
for (int row = 0; row < 4; row++) {
    for (int col = 0; col < 3; col++) {
        SetTableTextStyle(index, cell_array[row][col].cell_text_style);
        cell_text[index] = GetTableCellText(index);
        wcscpy
        cell_array[row][col].cell_text.str = (wchar_t*)cell_text[index];
        cell_array[row][col].cell_text.len = wcslen(cell_text[index]);
        cell_array[row][col].cell_image = 0;
        cell_array[row][col].cell_margin.left = 0;
        cell_array[row][col].cell_margin.right = 0;
        cell_array[row][col].cell_margin.bottom = 0;
        cell_array[row][col].cell_margin.top = 0;
        index++;
    }
}
float page_width = 0;
float page_height = 0;
FSDK_PDFPage_GetWidth(pdf_page, &page_width);
FSDK_PDFPage_GetHeight(pdf_page, &page_height);

FSTableData data;
data.row_count = 4;
data.col_count = 3;
data.outside_border_left.line_width = 1;
data.outside_border_left.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
data.outside_border_left.color = 0xFF000000;
data.outside_border_left.array_length_dashes = 0;
data.outside_border_top.line_width = 1;
data.outside_border_top.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
```

```

data.outside_border_top.color = 0xFF000000;
data.outside_border_top.array_length_dashes = 0;
data.outside_border_right.line_width = 1;
data.outside_border_right.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
data.outside_border_right.color = 0xFF000000;
data.outside_border_right.array_length_dashes = 0;
data.outside_border_bottom.line_width = 1;
data.outside_border_bottom.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
data.outside_border_bottom.color = 0xFF000000;
data.outside_border_bottom.array_length_dashes = 0;
data.inside_border_col.line_width = 1;
data.inside_border_col.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
data.inside_border_col.color = 0xFF000000;
data.inside_border_col.array_length_dashes = 0;
data.inside_border_row.line_width = 1;
data.inside_border_row.table_border_style = FSTableBorderStyle::e_FSTableBorderStyleSolid;
data.inside_border_row.color = 0xFF000000;
data.inside_border_row.array_length_dashes = 0;
data.merge_cells_length = 0;
data.row_height_array = NULL;
data.array_length_row_height = 0;
data.col_width_array = NULL;
data.array_length_col_width = 0;

FSRectF rect;
rect.left = 100;
rect.right = page_width - 100;
rect.top = page_height - 100;
rect.bottom = 550;
data.rect = rect;

FS_BOOL ret;
FSErrorCode error_code = FSDK_TableGenerator_AddTableToPage(pdf_page, data, (const
FSTableCellData**)cell_array, 4, 3, &ret);
if (error_code != e_FSErrSuccess) {
    printf("AddElectronicTable fail [%d]\r\n", error_code);
}

```

3.44 Accessibility

From version 8.4, Foxit PDF SDK supports to tag PDF files.

3.44.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'Accessibility' module permission in the license key

SDK Version: Foxit PDF SDK (C, C++, C#, Java, Python, Objective-C) 8.4 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

3.44.2 How to tag a PDF document

Foxit PDF SDK provides a taggedpdf demo located in the "\examples\simple_demo\taggedpdf" folder to show you how to use Foxit PDF SDK to tag a PDF document.

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_search_c.h"
#include "include/fs_convert_c.h"
#include "include/fs_taggedpdf_c.h"
...
FS_PDFDOC_HANDLE pdf_doc;
FSDK_PDFDoc_Create0(input_file, &pdf_doc);
FSDK_PDFDoc_Load(pdf_doc, NULL);
FS_TAGGEDPDF_HANDLE tagged_pdf;
error_code = FSDK_TaggedPDF_Create(pdf_doc, &tagged_pdf);
FS_PROGRESSIVE_HANDLE progressive;
FSDK_TaggedPDF_StartTagDocument(tagged_pdf, NULL, &progressive);
FSState progressive_state = e_FSToBeContinued;
while (progressive_state == e_FSToBeContinued)
    FSDK_Progressive_Continue(progressive, &progressive_state);
FS_BOOL return_value = false;
error_code = FSDK_PDFDoc_SaveAs(pdf_doc, output_file, 0, &return_value);
FSDK_TaggedPDF_Release(tagged_pdf);
FSDK_PDFDoc_Release(pdf_doc);
...
```

3.45 PDF to Office Conversion

Foxit PDF SDK provides APIs to convert PDF files to MS office suite formats while maintaining the layout and format of your original documents on Windows and Linux platforms.

3.45.1 System requirements

Platform: Windows, Linux

Programming Language: C, C++, Java, Python, C#, Node.js

License Key requirement: 'PDF2Office' module permission in the license key

SDK Version: Foxit PDF SDK for Windows (C, C++, Java, Python, C#) 9.0 or higher; Foxit PDF SDK for Linux (C, C++, Java, Python, C#) 9.1 or higher; Foxit PDF SDK (Node.js) 10.0 or higher

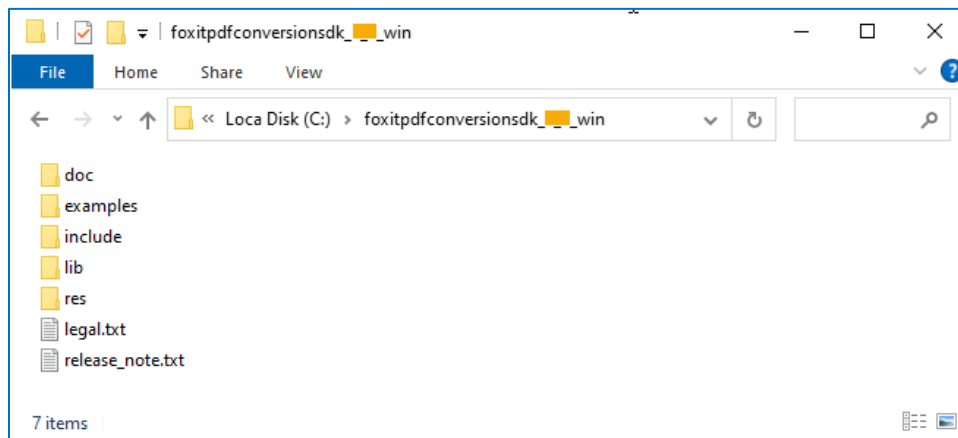
3.45.2 PDF to Office resource files

Please contact Foxit support team or sales team to get the PDF to Office resource files package naming Foxit PDF Conversion SDK (C++).

Note:

- From version 9.2, it requires Foxit PDF Conversion SDK 1.5 or higher.
- For version 9.0/9.1, it requires Foxit PDF Conversion SDK 1.4 or lower.

After getting Foxit PDF Conversion SDK package, extract it to a desired directory (for example, extract the package to a directory: "**foxitpdfconversionsdk*_win/**"), and then you can see the resource files for PDF to Office are as follows:



3.45.3 How to run the pdf2office demo

Foxit PDF SDK provides a pdf2office demo located in the "\\examples\\simple_demo\\pdf2office" folder to show you how to use Foxit PDF SDK to convert PDF files to MS office suite formats.

3.45.3.1 Prepare a PDF2Office resource directory

Before running the pdf2office demo, you should first extract the PDF to Office resource files (Foxit PDF Conversion SDK for C API) package to a desired directory (for example, extract the package to a directory: "**C:/foxitpdfconversionsdk*_win/**"), and then pass the engine file located in "lib" folder to the API **FSDK_PDF2Office_Initialize** to initialize PDF2Office engine.

3.45.3.2 Configure the demo

For pdf2office demo, you can configure the demo in the "`\examples\simple_demo\pdf2office\pdf2office.c`" file.

Specify the pdf2office engine directory

In the "pdf2office.c" file, add the path of the engine file "pdf2office" as follows, which will be used to convert PDF files to office files.

```
// Please ensure the path is valid.  
FSDK_PDF2Office_Initialize(L"C:/foxitpdfconversionsdk_*_win/lib/fpdfconversionsdk_win32.dll");
```

(Optional) Specify whether to enable machine learning-based recognition functionality

```
setting_data.enable_ml_recognition = FALSE;
```

(Optional) Specify the page range to be converted

```
setting_data.page_range = NULL;
```

(Optional) Specify whether to convert the comments in the PDF documents

```
setting_data.include_pdf_comments = TRUE;
```

(Optional) Specify whether to retain the page layout for PDF to Word conversion

```
setting_data.word_setting_data.enable_retain_page_layout = FALSE;
```

Note: Starting from version 10.1, the PDF to Office Conversion engine supports a timeout parameter. This parameter defines the maximum time allowed for the conversion process to complete. If the conversion exceeds the specified time, it will be terminated. The timeout must be a non-negative value. A value of 0 disables the timeout, allowing the conversion to proceed without any time limitation.

3.45.3.3 Run the demo

Once you run the demo successfully, the console will print the following by default:

```
Convert PDF file to Word format file with path.  
Convert PDF file to Word format file with stream.  
Convert PDF file to Excel format file with path.  
Convert PDF file to Excel format file with stream.  
Convert PDF file to PowerPoint format file with path.  
Convert PDF file to PowerPoint format file with stream.
```

The output files are located in "`\examples\simple_demo\output_files\pdf2office`" folder.

3.45.4 How to work with PDF2office API

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
#include "include/fx_stream_c.h"
#include "include/fs_pdf2office_c.h"

static FS_BOOL gNeedToPause(void* user_data) {
    return TRUE;
}

static void gProgressNotify(void* user_data, int converted_count, int total_count) {
}

FS_BOOL is_large_file = FALSE;
FILE* file = NULL;
FILE* filestream = NULL;
int ref = 0;
FS_INT64 cur_pos = 0;

FS_INT64 gGetSize(void* user_data) {
    if (!user_data) return 0;
    if (is_large_file) {
        long long size_long;
        _fseeki64(file, 0, SEEK_END);
        size_long = _ftelli64(file);
        return size_long;
    } else {
        fseek(file, 0, SEEK_END);
        return (FS_INT64)ftell(file);
    }
    return 0;
}

FSConvertCallback * callback_word = (FSConvertCallback*)malloc(sizeof(FSConvertCallback));
callback->user_data = callback;
callback->NeedToPause = gNeedToPause;
callback->ProgressNotify = gProgressNotify;
FS_PROGRESSIVE_HANDLE progressive_handle;
FSErrorCode error_code = FSDK_PDF2Office_StartConvertToWord(src_pdf_path, NULL, saved_word_file_path,
setting_data, callback_word, &progressive_handle);
if (error_code != e_FSErrSuccess) {
    switch (error_code) {
        case e_FSErrNoPDF2OfficeModuleRight:
            printf("[Failed] Conversion module is not contained in current Foxit PDF Conversion SDK keys.\n");
            break;
        default:
            printf("Exception: %d\n", error_code);
            break;
    }
    return 1;
}
```

```
int rate;
FSDK_Progressive_GetRateOfProgress(progressive_handle, &rate);
if (rate != 100) {
    FSState state = e_FSToBeContinued;
    while (e_FSToBeContinued == state) {
        FSDK_Progressive_Continue(progressive_handle, &state);
    }
}
printf("Convert PDF file to Word format file with path.\n");

FSDKConvertCallback * callback_excel = (FSDKConvertCallback*)malloc(sizeof(FSDKConvertCallback));
callback->user_data = callback;
callback->NeedToPause = gNeedToPause;
callback->ProgressNotify = gProgressNotify;
error_code = FSDK_PDF2Office_StartConvertToExcel(src_pdf_path, NULL, saved_excel_file_path, setting_data,
callback_excel, &progressive_handle);
if (error_code != e_FSErrSuccess) {
    switch (error_code) {
        case e_FSErrNoPDF2OfficeModuleRight:
            printf("[Failed] Conversion module is not contained in current Foxit PDF Conversion SDK keys.\n");
            break;
        default:
            printf("Exception: \n");
            break;
    }
    return 1;
}
FSDK_Progressive_GetRateOfProgress(progressive_handle, &rate);
if (rate != 100) {
    FSState state = e_FSToBeContinued;
    while (e_FSToBeContinued == state) {
        FSDK_Progressive_Continue(progressive_handle, &state);
    }
}
printf("Convert PDF file to Excel format file with path.\n");

FSDKConvertCallback * callback_powepoint = (FSDKConvertCallback*)malloc(sizeof(FSDKConvertCallback));
callback->user_data = callback;
callback->NeedToPause = gNeedToPause;
callback->ProgressNotify = gProgressNotify;
error_code = FSDK_PDF2Office_StartConvertToPowerPoint(src_pdf_path, NULL, saved_ppt_file_path,
setting_data, powepoint, &progressive_handle);
if (error_code != e_FSErrSuccess) {
    switch (error_code) {
        case e_FSErrNoPDF2OfficeModuleRight:
            printf("[Failed] Conversion module is not contained in current Foxit PDF Conversion SDK keys.\n");
            break;
        default:
            printf("Exception: %d\n",error_code);
            break;
    }
    return 1;
}
```



```
}
FSDK_Progressive_GetRateOfProgress(progressive_handle, &rate);
if (rate != 100) {
    FSState state = e_FSToBeContinued;
    while (e_FSToBeContinued == state) {
        FSDK_Progressive_Continue(progressive_handle, &state);
    }
}
printf("Convert PDF file to PowerPoint format file with path.\n");
```

3.46 DWG to PDF Conversion

From version 10.0, Foxit PDF SDK supports to convert DWG files to PDF files. If you want to use this feature, you should contact Foxit support team or sales team to get the engine files package.

3.46.1 System requirements

Platform: Windows, Linux (x86 and x64), Mac(x64)

Programming Language: C, C++, Java, C#, Python, Objective-C, Node.js

License Key requirement: 'DWG2PDF' module permission in the license key

SDK Version: Foxit PDF SDK 10.0 or higher

3.46.2 DWG To PDF engine files

Please contact Foxit support team or sales team to get the DWG to PDF engine files package.

After getting the package, extract it to the desired directory. For example, extract the package to a directory: "**D:/dwgtopdf/win**" for Windows.

3.46.3 How to run the dwg2pdf demo

Before running the dwg2pdf demo in the "\\examples\\simple_demo\\dwg2pdf" folder, you should first add the dwg2pdf engine file in the demo code, for example:

```
// "engine_path" is the path of the engine file "dwg2pdf" which is used to convert dwg to pdf. Please refer to
Developer Guide for more details.
const wchar_t* engine_path = L"D:/dwgtopdf/win";
```

Then, run the demo following the steps as the other demos.

3.46.4 How to convert DWG to PDF

```
#include "include/fs_basictypes_c.h"
#include "include/fs_convert_c.h"

// Make sure that SDK has already been initialized successfully.
...
```

```
FSHDWG2PDFSettingData pdf_setting_data;

pdf_setting_data.export_flags = e_FlagEmbeddedTTF; pdf_setting_data.export_hatches_type =
e_DWG2PDFExportHatchesTypeBitmap; pdf_setting_data.other_export_hatches_type =
e_DWG2PDFExportHatchesTypeBitmap; pdf_setting_data.gradient_export_hatches_type =
e_DWG2PDFExportHatchesTypeBitmap; pdf_setting_data.searchable_text_type =
e_DWG2PDFSearchableTextTypeNoSearch; pdf_setting_data.is_active_layout = false;
pdf_setting_data.paper_width = 640;
pdf_setting_data.paper_height = 900;

FSDK_Convert_FromDWG(engine_path.c_str(), dwg_file_path.c_str(), output_path.cstr(), pdf_setting_data);
...
```

3.47 OFD

OFD files, standing for Open Financial Document, are used for storing and exchanging digital financial documents. They are open and XML-based, making them specifically designed for financial documents like contracts, invoices, and statements.

OFD files contain structured data and graphical elements defining the document's layout and content, including text, images, vector graphics, annotations, and other related information. The XML format facilitates easy interpretation, manipulation, and rendering of the document's content.

OFD files provide various benefits, including document integrity, security, and interoperability. They can be digitally signed to ensure authenticity and can be encrypted to protect sensitive information. OFD files also support interactive features like form fields and digital signatures.

To work with OFD files, OFD viewer or editor software that supports the OFD standard is needed. These tools allow you to display, edit, convert, and print the contents of OFD documents.

In essence, OFD files provide a standardized and efficient method for representing financial documents digitally, simplifying the exchange, storage, and management of financial information.

3.47.1 System requirements

Platform: Windows, Linux (x64 and armv8)

Programming Language: C, C++, Java, C#, Python, Node.js

License Key requirement: 'OFD' module permission in the license key

SDK Version: Foxit PDF SDK 10.0 or higher

3.47.2 OFD engine file

Please contact Foxit support team or sales team to get the OFD engine file package.

After getting the package, extract it to the desired directory. For example, extract the package to a directory: "**D:/ofd/win**" for Windows.

3.47.3 How to run the ofd demo

Before running the ofd demo in the "\examples\simple_demo\ofd" folder, you should first add the ofd engine file in the demo code, for example:

```
// "engine_path" is the path of ofd engine file. Please refer to Developer Guide for more details.  
const char* engine_path = "D:/ofd/win/x64"; // For Windows x64
```

Then, run the demo following the steps as the other demos.

3.47.4 How to implement the conversion between OFD file and PDF file

```
#include "include/fs_common_c.h"  
#include "include/fs_convert_c.h"  
...  
// Initialize OFD engine.  
FSDK_Library_InitializeOFDEngine(engine_path);  
wchar_t src_ofd_path[MAX_FILE_PATH] = { 0 };  
swprintf_s(src_ofd_path, MAX_FILE_PATH, L"%swm_txttiled.ofd", input_path);  
  
wchar_t src_pdf_path[MAX_FILE_PATH] = { 0 };  
swprintf_s(src_pdf_path, MAX_FILE_PATH, L"%stest.pdf", input_path);  
wchar_t pdf2ofd_path[MAX_FILE_PATH] = { 0 };  
swprintf_s(pdf2ofd_path, MAX_FILE_PATH, L"%spdf2ofd.ofd", output_directory);  
wchar_t ofd2pdf_path[MAX_FILE_PATH] = { 0 };  
swprintf_s(ofd2pdf_path, MAX_FILE_PATH, L"%sofd2pdf.pdf", output_directory);  
FSOFDConvertParam convert_param;  
convert_param.is_embed_font = FALSE;  
// Convert OFD document to PDF document.  
FS_BOOL ret = FALSE;  
FSDK_Convert_FromOFD(src_ofd_path, L"", ofd2pdf_path, convert_param, &ret);  
// Convert PDF document to OFD document.  
FSDK_Convert_ToOFD(src_pdf_path, L"", pdf2ofd_path, convert_param, &ret);  
// Release OFD engine.  
FSDK_Library_ReleaseOFDEngine();
```

3.47.5 How to render OFD page

```
#include "include/fs_common_c.h"  
#include "include/fs_ofddoc_c.h"  
#include "include/fs_ofdpage_c.h"  
#include "include/fs_ofdrenderer_c.h"  
  
// Initialize OFD engine.  
FSDK_Library_InitializeOFDEngine(engine_path);  
  
// Render OFD document to bitmap.
```

```

{
    wchar_t render_file_path[MAX_FILE_PATH] = { 0 };
    swprintf_s(render_file_path, MAX_FILE_PATH, L"%scontent_flag.ofd", input_path);
    FS_OFDDOC_HANDLE doc;
    FSDK_OFDDoc_Create0(render_file_path, L"", &doc);
    FS_OFDPAGE_HANDLE ofd_page;
    FSDK_OFDDoc_GetPage(doc, 0, &ofd_page);
    // Get the size of the page.
    float w = 0;
    FSDK_OFDPage_GetWidth(ofd_page, &w);
    float h = 0;
    FSDK_OFDPage_GetHeight(ofd_page, &h);
    FS_BITMAP_HANDLE bitmap;
    FSDK_Bitmap_Create(w, h, e_FSDIBArgb, NULL, 0, &bitmap);
    FS_RECT fRect;
    fRect.left = 0;
    fRect.top = h;
    fRect.right = w;
    fRect.bottom = 0;
    FSDK_Bitmap_FillRect(bitmap, 0xFFFFFFFF, &fRect);
    // Get the display matrix of the page.
    FSMATRIX matrix_1;
    FSDK_OFDPage_GetDisplayMatrix(ofd_page, 0, 0, w, h, e_FSRotation0, &matrix_1);

    FS_OFDRENDERER_HANDLE ofd_render;
    FSDK_OFDRenderer_Create0(bitmap, &ofd_render);

    FS_PROGRESSIVE_HANDLE progressive;
    FSDK_OFDRenderer_StartRender(ofd_render, ofd_page, matrix_1, &progressive);

    Save2Image(bitmap);
    // Add the bitmap to image and save the image.
    FS_IMAGE_HANDLE image;
    FSDK_Image_Create(&image);
    FS_BOOL ret;
    FSDK_Image_AddFrame(image, bitmap, &ret);
    wchar_t sSaveFilePath[MAX_FILE_PATH] = { 0 };
    swprintf_s(sSaveFilePath, MAX_FILE_PATH, L"%srenderBitmap.bmp", output_directory);
    FSDK_Image_SaveAs(image, sSaveFilePath, &ret);
    FSDK_OFDRenderer_Release(ofd_render);
    FSDK_OFDPAGE_Release(ofd_page);
    FSDK_OFDDoc_Release(doc);
}
// Release OFD engine.
FSDK_Library_ReleaseOFDEngine();

```

3.48 Paragraph Editing

Foxit PDF SDK offers a versatile set of tools for developers to fine-tune and customize text in PDF documents. The paragraph editing module provides complex adjustments, joining, and splitting

functions that allow users to have precise control over the content of the document. The features are complemented by an intuitive UI implementation that facilitates efficient editing, ensuring a seamless and customized experience for managing text paragraphs.

The paragraph editing functionality revolves around two core modules, the **ParagraphEditing** module, and the **JoinSplit** module.

The **ParagraphEditing** module is designed to offer a variety of text editing operations, enabling users to easily perform the following actions according to their specific requirements:

- **Insert Text:** Insert new content at specific locations, allowing for customization of the document's precise layout.
- **Delete Text:** Delicately remove paragraphs or characters, enabling highly customized content trimming.
- **Modify Text:** Adjust existing text, including its content and formatting, to suit different editing styles.
- **Format Adjustment:** Support fine adjustments to paragraph formats and text styles, allowing for more accurate typesetting.

The **JoinSplit** module contains four vital operation types to support more complex text processing requirements:

- **Join:** Integrate multiple text blocks, enhancing content layout and overall document consistency.
- **Split:** Finely split text blocks, providing flexibility to manage various sections of the document.
- **Link:** Establish connections between text blocks, ensuring consistency in associated content.
- **Unlink:** Disconnect links between text blocks, offering more control over editing.

3.48.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C

License Key requirement: 'AdvEdit' module permission in the license key

SDK Version: Foxit PDF SDK 10.0 or higher

3.48.2 How to work with paragraph editing

```
#include "include/fs_basictypes_c.h"  
#include "include/fs_common_c.h"
```

```
#include "include/fs_pdfdoc_c.h"
#include "include/fs_pdfpage_c.h"
#include "include/fs_paragraphediting_c.h"
...

FSParagraphEditingProviderCallback* provider_callback;

FSDK_PDFPage_StartParse(page, e_FSParseFlagsParsePageNormal, NULL, FALSE, &return_value);
FSDK_PDFPage_GetHeight(page, &floatheight);
FSDK_PDFPage_GetIndex(page, &page_index_);

provider_callback = (FSParagraphEditingProviderCallback*)malloc(sizeof(FSParagraphEditingProviderCallback));
provider_callback->user_data = (void*)provider_callback;
provider_callback->Release = gRelease;
provider_callback->GetRenderMatrix = gGetRenderMatrix;
provider_callback->GetPageViewHandle = gGetPageViewHandle;
provider_callback->GetClientRect = gGetClientRect;
provider_callback->GetScale = gGetScale;
provider_callback->GotoPageView = gGotoPageView;
provider_callback->GetVisiblePageIndexArray = gGetVisiblePageIndexArray;
provider_callback->GetPageVisibleRect = gGetPageVisibleRect;
provider_callback->GetPageRect = gGetPageRect;
provider_callback->GetCurrentPageIndex = gGetCurrentPageIndex;
provider_callback->GetRotation = gGetRotation;
provider_callback->InvalidateRect = gInvalidateRect;
provider_callback->AddUndoItem = gAddUndoItem;
provider_callback->SetDocChangeMark = gSetDocChangeMark;
provider_callback->NotifyTextInputReachLimit = gNotifyTextInputReachLimit;
FSDK_ParagraphEditingMgr_Create(provider_callback, doc, &provider_mgr);

// Paragraph_editing
{
    FSDK_ParagraphEditingMgr_GetParagraphEditing(provider_mgr, &provider_editing);
    FSDK_ParagraphEditing_Activate(provider_editing, &return_value);
    point.x = 95;
    point.y = floatheight - 728;
    FSDK_ParagraphEditing_StartEditing(provider_editing, 0, point, point);
    FSDK_ParagraphEditing_SetFontSize(provider_editing, 24);
    FSDK_ParagraphEditing_SetUnderline(provider_editing, TRUE);
    FSDK_ParagraphEditing_InsertText(provider_editing, L"InsertText_Paragraph_editing", &return_value);
    FSDK_ParagraphEditing_Deactivate(provider_editing, &return_value);
    swprintf_s(output_file, MAX_FILE_PATH, L"%lsParagraph_editing.pdf", output_directory);
    error_code = FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &return_result);
    if (error_code != e_FSErrSuccess) {
        printf("Save doc error!\n");
        FSDK_PDFDoc_Release(doc);
        return 1;
    }
}

// Join&split
{
```

```

point.x = 289;
point.y = 659;
FSDK_ParagraphEditingMgr_GetJoinSplit(provider_mgr, &join_split);
FSDK_ParagraphEditing_Activate(join_split, &return_value);
FSDK_JoinSplit_OnLButtonDown(join_split, 0, point, &return_value);
FSDK_JoinSplit_OnLButtonUp(join_split, 0, point, &return_value);
FSDK_JoinSplit_SplitBoxes(join_split);
FSDK_JoinSplit_Deactivate(join_split, &return_value);
swprintf_s(output_file, MAX_FILE_PATH, L"%lsSplit_Boxes.pdf", output_directory);
error_code = FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &return_result);
if (error_code != e_FSErrSuccess) {
    printf("Save doc error!\n");
    FSDK_PDFDoc_Release(doc);
    return 1;
}

FSDK_ParagraphEditing_Activate(join_split, &return_value);
point.x = 307;
point.y = floatheight - 637;
FSDK_JoinSplit_OnLButtonDown(join_split, 0, point, &return_value);
FSDK_JoinSplit_OnLButtonUp(join_split, 0, point, &return_value);
point.x = 307;
point.y = floatheight - 453;
FSDK_JoinSplit_OnLButtonDown(join_split, 0, point, &return_value);
FSDK_JoinSplit_OnLButtonUp(join_split, 0, point, &return_value);
FSDK_JoinSplit_JoinBoxes(join_split);
FSDK_JoinSplit_Deactivate(join_split, &return_value);
swprintf_s(output_file, MAX_FILE_PATH, L"%lsJoin_Boxes.pdf", output_directory);
error_code = FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagNoOriginal, &return_result);
}

```

3.49 3D Rendering

3D Rendering in PDFs helps convert 3D models into 2D images or animations, which can be embedded in the document. The 3D models can be created using specialized 3D modeling software or CAD (Computer-Aided Design) software and can then be rendered into a 2D format that can be easily viewed within a PDF.

3D Annotation is a feature that allows users to add contextual information to specific parts of these 3D models within the PDF. The annotations can include text, images, and even links to external resources, providing more detailed information and insights about the model.

3.49.1 System requirements

Platform: Windows

Programming Language: C, C++, Java, C#, Python

License Key requirement: '3D' module permission in the license key

SDK Version: Foxit PDF SDK 10.0 or higher

3.49.2 How to display the 3D annotation

```
#include "include/fs_pdf3d_c.h"

// Load PDF document.
...

FS_PDF3DCONTEXT_HANDLE pdf_context;
FSDK_PDF3DContext_Create(pdf_doc, &pdf_context);

// Get the 3d annotation instance array.
FS_PDF3DANNOTINSTANCE_HANDLE* annot_data_arr = NULL;
FS_UINT32 array_length;
FSDK_PDF3DContext_GetPage3DAnnotArray(pdf_context, 0, NULL, &array_length);
if (array_length == 0) return;
annot_data_arr = (FS_PDF3DANNOTINSTANCE_HANDLE*)malloc(sizeof(FS_PDF3DANNOTINSTANCE_HANDLE) *
array_length);
FSDK_PDF3DContext_GetPage3DAnnotArray(pdf_context, 0, annot_data_arr, &array_length);

// Class parameter.
FS_PDF3DANNOTINSTANCE_HANDLE annotData = annot_data_arr[0];
// Activate the canvas to display the 3d annotation. Pass in a window handle to embed canvas.
FSDK_PDF3DAnnotInstance_ActivateCanvas(annotData, window handle);
```

3.49.3 How to set render mode and controller

```
#include "include/fs_pdf3d_c.h"

// Rotate to view 3D annotations.
FSDK_PDF3DAnnotInstance_SetController(FSPDF3DController::e_FSControllerRotate);

// Render 3D annotations as transparent.
FSDK_PDF3DAnnotInstance_SetRenderMode(FSPDF3DRenderMode::e_FSRenderModeTransparent);
```


FAQ

1. How to fix the "'xcopy' exited with code 9009" error when building demos in Visual Studio?

When building demos in Visual Studio, if you encounter the error "'xcopy' exited with code 9009" as follows:

```
'xcopy ..\..\..\..\lib\gsdk_sn.txt ..\..\..\ /y > null
xcopy ..\..\..\..\lib\gsdk_key.txt ..\..\..\ /y > null
xcopy ..\..\..\..\lib\$(PlatformName)_vc10\fsdk.dll ..\..\..\ /y > null
xcopy ..\..\..\..\lib\$(PlatformName)_vc10\fsdk_dotnet.dll ..\..\..\ /y > null' exited with code 9009
```

Please check the following points:

- 1) Check whether the **xcopy.exe** is in the "%SystemRoot%\System32" directory, if not, copy one from another machine.
- 2) Check whether the system **PATH** environment variables have been set correctly. It should contain "%SystemRoot%\System32;%SystemRoot%;", if the environment variables for **xcopy** is right, but it still reports the error, please put the path of **xcopy** in front of others. Maybe some other environment variables have spell mistakes, so that cause the subsequent environment variables are invalid. Please check it.

After checking, open a command prompt, type xcopy command, if it can be recognized, close Visual Studio, and restart the demos. The error should be fixed.

2. How do I get text objects in a specified position of a PDF and change the contents of the text objects?

To get text objects in a specified position of a PDF and change the contents of the text objects using Foxit PDF SDK, you can follow the steps below:

- 1) Open a PDF file.
- 2) Load PDF pages and get the page objects.
- 3) Use **FSDK_PDFPage_GetGraphicsObjectsAtPoint** to get the text object at a certain position. Note: use the page object to get rectangle to see the position of the text object.
- 4) Change the contents of the text objects and save the PDF document.

Following is the sample code:

```
#include "include/fs_basictypes_c.h"
#include "include/fs_common_c.h"
```

```
#include "include/fs_pdfdoc.h"
#include "include/fs_pdfgraphicsobject_c.h"

...

bool ChangeTextObjectContent()
{
    wchar_t input_file[MAX_FILE_PATH];
    swprintf_s(input_file, MAX_FILE_PATH, L"%lsAboutFoxit.pdf", input_path);
    FS_PDFDOC_HANDLE doc;
    FSDK_PDFDoc_Create0(input_file, &doc);
    FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
    if (error_code != e_FSErrSuccess) {
        FSDK_PDFDoc_Release(doc);
        wprintf(L"The Doc [%ls] Error: %d\n", input_file, &error_code);
        return false;
    }
    // Get original shading objects from the first PDF page.
    FS_PDFPAGE_HANDLE original_page;
    FSDK_PDFDoc_GetPage(doc, 0, &original_page);
    FS_PROGRESSIVE_HANDLE progressive;
    FSDK_PDFPage_StartParse(original_page, e_FSParseFlagsParsePageNormal, NULL, false, &progressive);
    FSPointF pointf;
    pointf.x = 92;
    pointf.y = 762;
    FS_UINT32 arr_length = 0;
    FSDK_PDFPage_GetGraphicsObjectsAtPoint(original_page, pointf, 10, e_FSTypeText, NULL,
    &arr_length);
    FS_GRAPHICSOBJECT_HANDLE*arr = malloc(sizeof(FS_GRAPHICSOBJECT_HANDLE) * arr_length);
    FSDK_PDFPage_GetGraphicsObjectsAtPoint(original_page, pointf, 10, e_FSTypeText, arr, &arr_length);
    for(int i = 0; i<arr_length; i++)
    {
        FS_GRAPHICSOBJECT_HANDLE graphobj = arr[i];
        FS_TEXTOBJECT_HANDLE textobj;
        FSDK_GraphicsObject_GetTextObject(graphobj, &textobj);
        FSDK_TextObject_SetText(textobj, L"Foxit Test");
    }
    FS_BOOL return_result = false;
    FSDK_GraphicsObjects_GenerateContent(original_page, &return_result);
    wchar_t output_directory[MAX_FILE_PATH];
    wchar_t output_file[MAX_FILE_PATH];
    swprintf_s(output_directory, MAX_FILE_PATH, L"%lsgraphics_objects/", output_path)
    swprintf_s(output_directory, MAX_FILE_PATH, L"%lsAfter_revise.pdf", output_directory)
    FSDK_PDFDoc_SaveAs(doc, output_file, e_FSSaveFlagsSaveFlagNormal, &return_result);
    return true;
}
```

3. Can I change the DPI of an embedded TIFF image?

No, you cannot change it. The DPI of the images in PDF files is static, so if the images already exist, Foxit PDF SDK does not have functions to change its DPI.

The solution is that you can use third-party library to change the DPI of an image, and then add it to the PDF file.

Note: Foxit PDF SDK provides a function "FSDK_Image_SetDPIs" which can set the DPI property of an image object. However, it only supports the images that are created by Foxit PDF SDK or created by function "FSDK_Image_AddFrame", and it does not support the image formats of JPX, GIF and TIF.

4. Why do I encounter "Fail to initialize the engine file or cannot load the engine file" for OCR and DWG2PDF modules on Windows 7 when running the corresponding simple demos, even if the engine files have been upgraded to the latest and the simple demos have configured the engine path correctly?

For Windows 7, you need to copy the dll files starting with **api-ms-win*** and the **ucrtbase.dll** in the engine directory to the system directory.

- If you are using a 32-bit engine and running on a 32-bit system, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/System32".
- If you are using a 32-bit engine and running on a 64-bit system, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/SysWOW64".
- If you are using a 64-bit engine, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/System32".

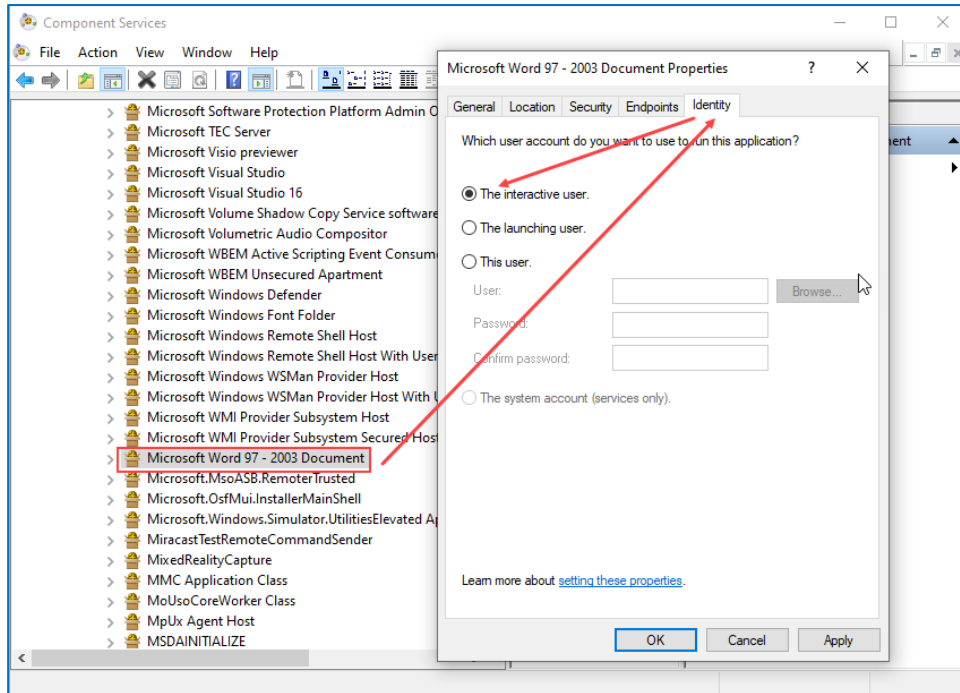
5. How to run the Office2PDF functionality in Windows services?

To run the Office2PDF functionality in Windows services, you need to configure the Office Component Services and permissions.

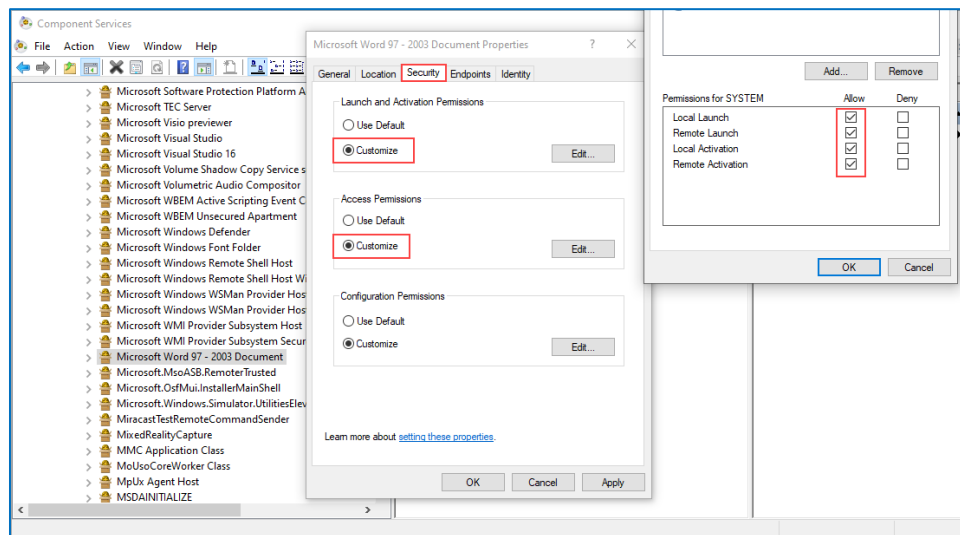
Take the Word component as an example.

- 1) Press **Win+R**, and then type **Dcomcnfg** to open Component Services, find [Component Services] -> [Computers] -> [My Computer] -> [DCOM Config] -> [Microsoft Word 97-2003 Document], right-click it and select Properties. Choose [Identity], and set it to "**The interactive user**".

Note: if you can't find [Microsoft Word 97-2003 Document] with **Dcomcnfg** command, you can try to use the "**comexp.msc -32**" command.



- 2) Set permissions. click [Security], set the [Launch and Activation Permissions] and [Access Permissions] to **Customize**. Click **Edit**, and add the current login account of the system and enable all permissions.



- 3) After finishing the above settings, the Word2PDF functionality can be run in the Windows services.

APPENDIX

Supported JavaScript List

Objects' property or method

Object	Properties/Method Names	Minimum Supported SDK Version
annotation properties	alignment	V7.0
	author	V7.0
	contents	V7.0
	creationDate	V7.0
	fillColor	V7.0
	hidden	V7.0
	modDate	V7.0
	name	V7.0
	opacity	V7.0
	page	V7.0
	readOnly	V7.0
	rect	V7.0
	richContents	V7.1
	rotate	V7.0
	strokeColor	V7.0
	textSize	V7.0
	type	V7.0
	AP	V9.0
	arrowBegin	V9.0
	arrowEnd	V9.0
	attachIcon	V9.0
	attachment	V9.0
	borderEffectIntensity	V9.0
	borderEffectStyle	V9.0
	callout	V9.0
	caretSymbol	V9.0
	dash	V9.0
	delay	V9.0
	doc	V9.0
	doCaption	V9.0
	gestures	V9.0
	inReplyTo	V9.0
	intent	V9.0
	leaderExtend	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	leaderLength	V9.0
	lineEnding	V9.0
	lock	V9.0
	notelcon	V9.0
	noView	V9.0
	point	V9.0
	points	V9.0
	popupOpen	V9.0
	popupRect	V9.0
	print	V9.0
	quads	V9.0
	refType	V9.0
	richDefaults	V9.0
	seqNum	V9.0
	soundIcon	V9.0
	style	V9.0
	subject	V9.0
	textFont	V9.0
	toggleNoView	V9.0
	vertices	V9.0
	width	V9.0
annotation method	destroy	V7.0
	getProps	V9.0
	setProps	V9.0
	getStateInModel	V9.0
app properties	activeDocs	V4.0
	calculate	V4.0
	formsVersion	V4.0
	fs	V4.0
	fullscreen	V4.0
	language	V4.2
	platform	V4.0
	runtimeHighlight	V4.0
	viewerType	V4.0
	viewerVariation	V4.0
	viewerVersion	V4.0
	printerNames	V8.4
	runtimeHighlightColor	V8.4
	constants	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
app methods	alert	V4.0
	beep	V4.0
	browseForDoc	V4.0
	clearInterval	V4.0
	clearTimeout	V4.0
	launchURL	V4.0
	mailMsg	V4.0
	response	V4.0
	setInterval	V4.0
	setTimeout	V4.0
	popupMenu	V4.0
	execDialog	V8.4
	execMenuItem	V8.4
	newDoc	V8.4
	openDoc	V8.4
	popupMenuEx	V8.4
	addMenuItem	V8.4
	addSubMenu	V8.4
	addToolButton	V8.4
	removeToolButton	V8.4
	listMenuItems	V8.4
	trustedFunction	V8.4
	beginPriv	V8.4
	endPriv	V8.4
color properties	black	V4.0
	blue	V4.0
	cyan	V4.0
	dkGray	V4.0
	gray	V4.0
	green	V4.0
	ltGray	V4.0
	magenta	V4.0
	red	V4.0
	transparent	V4.0
	white	V4.0
	yellow	V4.0
color methods	convert	V4.0
	equal	V4.0
document properties	author	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	baseURL	V4.0
	bookmarkRoot	V7.0
	calculate	V4.0
	Collab	V4.0
	creationDate	V4.0
	creator	V4.0
	delay	V4.0
	dirty	V4.0
	documentFileName	V4.0
	external	V4.0
	filesize	V4.0
	icons	V4.0
	info	V4.0
	keywords	V4.0
	modDate	V4.0
	numFields	V4.0
	numPages	V4.0
	pageNum	V4.0
	path	V4.0
	producer	V4.0
	subject	V4.0
	title	V4.0
	URL	V8.4
	dataObjects	V8.4
	hostContainer	V8.4
	templates	V8.4
	media	V8.4
	dynamicXFAForm	V8.4
	mouseX	V8.4
	mouseY	V8.4
	pageWindowRect	V8.4
	securityHandler	V8.4
	zoom	V8.4
	zoomType	V8.4
	layout	V8.4
	xfa	V8.4
document methods	addAnnot	V7.0
	addField	V4.0
	addIcon	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	calculateNow	V4.0
	deletePages	V4.0
	exportAsFDF	V4.0
	flattenPages	V7.1
	getAnnot	V7.0
	getAnnots	V7.0
	getField	V4.0
	getIcon	V4.0
	getNthFieldName	V4.0
	getOCGs	V4.0
	getPageBox	V4.0
	getPageNthWord	V4.0
	getPageNthWordQuads	V4.0
	getPageNumWords	V4.0
	getPageRotation	V7.0
	getPrintParams	V4.0
	getURL	V4.0
	importAnFDF	V4.0
	insertPages	V6.2
	mailForm	V4.0
	print	V4.0
	removeField	V4.0
	replacePages	V6.2
	resetForm	V4.0
	submitForm	V4.0
	mailDoc	V4.0
	addWatermarkFromFile	V8.4
	addWatermarkFromText	V8.4
	getPageLabel	V8.4
	setPageLabels	V8.4
	gotoNamedDest	V8.4
	saveAs	V8.4
	scroll	V8.4
	setPageTabOrder	V8.4
	selectPageNthWord	V8.4
	syncAnnotScan	V8.4
	getAnnot3D	V8.4
	getAnnots3D	V8.4
	addLink	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	removeLinks	V8.4
	getLinks	V8.4
	importIcon	V8.4
	removeIcon	V8.4
	addWeblinks	V8.4
	removeWeblinks	V8.4
	closeDoc	V8.4
	exportDataObject	V8.4
	importDataObject	V8.4
	removeDataObject	V8.4
	getDataObject	V8.4
	embedDocAsDataObject	V8.4
	createTemplate	V8.4
	removeTemplate	V8.4
	getTemplate	V8.4
	exportAsText	V8.4
	importTextData	V8.4
	exportAsXFDF	V8.4
	importAnXFDF	V8.4
	exportAsXFDFStr	V8.4
	extractPages	V8.4
	movePage	V8.4
	newPage	V8.4
	getOCGOrder	V8.4
	setOCGOrder	V8.4
	setPageBoxes	V8.4
	setPageRotations	V8.4
	setPageTransitions	V9.1
	getPageTransition	V9.1
event properties	change	V4.0
	changeEx	V4.0
	commitKey	V4.0
	fieldFull	V4.0
	keyDown	V4.0
	modifier	V4.0
	name	V4.0
	rc	V4.0
	selEnd	V4.0
	selStart	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	shift	V4.0
	source	V4.0
	target	V4.0
	targetName	V4.0
	type	V4.0
	value	V4.0
	willCommit	V4.0
event methods	add	V9.0
field properties	alignment	V4.0
	borderStyle	V4.0
	buttonAlignX	V4.0
	buttonAlignY	V4.0
	buttonFitBounds	V4.0
	buttonPosition	V4.0
	buttonScaleHow	V4.0
	buttonScaleWhen	V4.0
	calcOrderIndex	V4.0
	charLimit	V4.0
	comb	V4.0
	commitOnSelChange	V4.0
	currentValueIndices	V4.0
	defaultValue	V4.0
	doNotScroll	V4.0
	doNotSpellCheck	V4.0
	delay	V4.0
	display	V4.0
	doc	V4.0
	editable	V4.0
	exportValues	V4.0
	hidden	V4.0
	fileSelect	V4.0
	fillColor	V4.0
	lineWidth	V4.0
	highlight	V4.0
	multiline	V4.0
	multipleSelection	V4.0
	name	V4.0
	numItems	V4.0
	page	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	password	V4.0
	print	V4.0
	radiosInUnison	V4.0
	readonly	V4.0
	rect	V4.0
	required	V4.0
	richText	V4.0
	rotation	V4.0
	strokeColor	V4.0
	style	V4.0
	textColor	V4.0
	textFont	V4.0
	textSize	V4.0
	type	V4.0
	userName	V4.0
	value	V4.0
	valueAsString	V4.0
	richValue	V9.0
	submitName	V9.0
field methods	browseForFileToSubmit	V4.0
	buttonGetCaption	V4.0
	buttonGetIcon	V4.0
	buttonSetCaption	V4.0
	buttonSetIcon	V4.0
	checkThisBox	V4.0
	clearItems	V4.0
	defaultIsChecked	V4.0
	deleteItemAt	V4.0
	getArray	V4.0
	getItemAt	V4.0
	insertItemAt	V4.0
	isBoxChecked	V4.0
	isDefaultChecked	V4.0
	setAction	V4.0
	setFocus	V4.0
	setItems	V4.0
	buttonImportIcon	V9.0
	getLock	V9.0
	setLock	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	signatureGetModifications	V9.0
	signatureGetSeedValue	V9.0
	signatureInfo	V9.0
	signatureSetSeedValue	V9.0
	signatureSign	V9.0
	signatureValidate	V9.0
global methods	setPersistent	V4.0
Icon properties	name	V4.0
util methods	printd	V4.0
	printf	V4.0
	printx	V4.0
	scand	V4.0
	iconStreamFromIcon	V9.0
identity properties	loginName	V4.2
	name	V4.2
	corporation	V4.2
	email	V4.2
collab properties	user	V6.2
ocg properties	name	V6.2
ocg methods	setAction	V6.2
bookmark properties	color	V8.4
	open	V8.4
	name	V8.4
	parent	V8.4
	children	V8.4
	language	V8.4
	style	V8.4
	platform	V8.4
bookmark methods	createChild	V8.4
	insertChild	V8.4
	execute	V8.4
	setAction	V8.4
	remove	V8.4
certificate properties	binary	V8.4
	issuerDN	V8.4
	keyUsage	V8.4
	MD5Hash	V8.4
	privateKeyValidityEnd	V8.4
	privateKeyValidityStart	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	SHA1Hash	V8.4
	serialNumber	V8.4
	subjectCN	V8.4
	subjectDN	V8.4
	validityEnd	V8.4
	validityStart	V8.4
RDN properties	c	V8.4
	cn	V8.4
	e	V8.4
	l	V8.4
	o	V8.4
	ou	V8.4
	st	V8.4
security properties	handlers	V9.0
security methods	getHandler	V9.0
	importFromFile	V9.0
securityHandler properties	appearances	V9.0
	isLoggedIn	V9.0
	loginName	V9.0
	loginPath	V9.0
	name	V9.0
	uiName	V9.0
securityHandler methods	login	V9.0
	logout	V9.0
	newUser	V9.0
signatureInfo properties	objValidity	V9.0
	idValidity	V9.0
	idPrivValidity	V9.0
	docValidity	V9.0
	byteRange	V9.0
	verifyHandlerUIName	V9.0
	verifyHandlerName	V9.0
	verifyDate	V9.0
	subFilter	V9.0
	statusText	V9.0
	status	V9.0
	reason	V9.0
	name	V9.0
	mdp	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	location	V9.0
	handlerUIName	V9.0
	handlerUserName	V9.0
	handlerName	V9.0
	dateTrusted	V9.0
	date	V9.0
search properties	attachments	V9.0
	bookmarks	V9.0
	docText	V9.0
	ignoreAccents	V9.0
	markup	V9.0
	matchCase	V9.0
	matchWholeWord	V9.0
	maxDocs	V9.0
	proximity	V9.0
	stem	V9.0
	wordMatching	V9.0
	ignoreAsianCharacterWidth	V9.0
search methods	query	V9.0
	addIndex	V9.0
	removeIndex	V9.0
link properties	borderColor	V8.4
	borderWidth	V8.4
	highlightMode	V8.4
	rect	V8.4
link methods	setAction	V8.4
app.media properties	align	V8.4
	canResize	V8.4
	ifOffScreen	V8.4
	over	V8.4
	windowType	V8.4
app.media methods	createPlayer	V8.4
	openPlayer	V8.4
doc.media methods	getOpenPlayers	V8.4
Playerargs properties	doc	V8.4
	annot	V8.4
	rendition	V8.4
	URL	V8.4
	mimeType	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	settings	V8.4
	events	V8.4
MediaPlayer properties	isOpen	V8.4
	isPlaying	V8.4
	settings	V8.4
	visible	V8.4
MediaPlayer methods	close	V8.4
	play	V8.4
	seek	V8.4
	stop	V8.4
MediaSettings properties	autoPlay	V8.4
	baseUrl	V8.4
	bgColor	V8.4
	bgOpacity	V8.4
	duration	V8.4
	floating	V8.4
	page	V8.4
	repeat	V8.4
	showUI	V8.4
	visible	V8.4
	volume	V8.4
	windowType	V8.4
floating properties	align	V8.4
	over	V8.4
	canResize	V8.4
	hasClose	V8.4
	hasTitle	V8.4
	title	V8.4
	ifOffScreen	V8.4
	rect	V8.4
eventListener methods	afterClose	V9.0
	afterPlay	V9.0
	afterReady	V9.0
	afterSeek	V9.0
	afterStop	V9.0
	onClose	V9.0
	onPlay	V9.0
	onReady	V9.0
	onSeek	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	onStop	V9.0
Template properties	hidden	V9.1
	name	V9.1
Template method	spawn	V9.1
span properties	alignment	V9.1
	fontFamily	V9.1
	fontStretch	V9.1
	fontWeight	V9.1
	fontStyle	V9.1
	strikethrough	V9.1
	subscript	V9.1
	superscript	V9.1
	text	V9.1
	textColor	V9.1
	textSize	V9.1
	underline	V9.1
soap properties	wireDump	V9.1
Soap method	request	V9.1
	streamDigest	V9.1
	streamEncode	V9.1
	streamFromString	V9.1
	stringFromStream	V9.1
hostContainer method	postMessage	V9.2
Fullscreen properties	transitions	V9.2
	defaultTransition	V9.2
	loop	V9.2
	timeDelay	V9.2
	useTimer	V9.2
	isFullScreen	V9.2

Global methods

Method Names	Minimum Supported SDK Version
AFNumber_Format	V4.0
AFNumber_Keystroke	V4.0
AFPercent_Format	V4.0
AFPercent_Keystroke	V4.0
AFDate_FormatEx	V4.0
AFDate_KeystrokeEx	V4.0
AFDate_Format	V4.0

Method Names	Minimum Supported SDK Version
AFDate_Keystroke	V4.0
AFTime_FormatEx	V4.0
AFTime_KeystrokeEx	V4.0
AFTime_Format	V4.0
AFTime_Keystroke	V4.0
AFSpecial_Format	V4.0
AFSpecial_Keystroke	V4.0
AFSpecial_KeystrokeEx	V4.0
AFSimple	V4.0
AFMakeNumber	V4.0
AFSimple_Calculate	V4.0
AFRange_Validate	V4.0
AFMergeChange	V4.0
AFParseDateEx	V4.0
AFExtractNums	V4.0

REFERENCES

[1] PDF reference 1.7

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51502

[2] PDF reference 2.0

<https://www.iso.org/standard/63534.html>

[3] Foxit PDF SDK API reference

sdk_folder/doc/Foxit PDF SDK API Reference.html

Note: sdk_folder is the directory of unzipped package.

SUPPORT

Foxit Support

In order to provide you with a more personalized support for a resolution, please log in to your [Foxit account](#) and submit a ticket so that we can collect details about your issue. We will work to get your problem solved as quickly as we can once your ticket is routed to our support team.

You can also check out our [Support Center](#), choose Foxit PDF SDK which also has a lot of helpful articles that may help with solving your issue.

Phone Support:

Phone: 1-866-MYFOXIT or 1-866-693-6948