

Developer Guide

Foxit PDF SDK

for C API

Microsoft® Partner
Gold Independent Software Vendor (ISV)

©Foxit Software Incorporated. All rights reserved.

TABLE OF CONTENTS

1. Introduction to Foxit PDF SDK 1

1.1 Why Foxit PDF SDK is your choice 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 16

3.2.7 How to save a document into memory buffer by WriterCallback 16

3.3 Page 17

3.3.1 How to get page size 17

3.3.2 How to calculate bounding box of page contents 18

3.3.3 How to create a PDF page and set the size 18

3.3.4 How to delete a PDF page 18
3.3.5 How to flatten a PDF page 18
3.3.6 How to get and set page thumbnails in a PDF document 19
3.4 Render 19
3.4.1 How to render a page to a bitmap 20
3.4.2 How to render page and annotation 21
3.5 Attachment 21
3.5.1 How to export the embedded attachment file from a PDF and save it as a single file 21
3.5.2 How to remove all the attachments of a PDF 22
3.6 Text Page 23
3.6.1 How to extract text from a PDF page 23
3.6.2 How to get the text within a rectangle area in a PDF 24
3.7 Text Search 24
3.7.1 How to search a text pattern in a PDF 24
3.8 Text Link 25
3.8.1 How to retrieve hyperlinks in a PDF page 26
3.9 Bookmark 26
3.9.1 How to find and list all bookmarks of a PDF 26
3.9.2 How to insert a new bookmark 27
3.9.3 How to create a table of contents based on bookmark information in PDFs 28
3.10 Form (AcroForm) 29
3.10.1 How to load the forms in a PDF 29
3.10.2 How to count form fields and get/set the properties 30
3.10.3 How to export the form data in a PDF to a XML file 30
3.10.4 How to import form data from a XML file 31
3.10.5 How to get coordinates of a form field 31
3.11 XFA Form 32
3.11.1 How to load XFADoc and represent an Interactive XFA form 32

3.11.2 How to export and import XFA form data 33
3.12 Form Design 33
3.12.1 How to add a text form field to a PDF 34
3.12.2 How to remove a text form field from a PDF 35
3.13 Annotations 35
3.13.1 General 35
3.13.2 Import annotations from or export annotations to a FDF file 42
3.14 Image Conversion 43
3.14.1 How to convert PDF pages to bitmap files 43
3.14.2 How to convert an image file to PDF file 44
3.15 Watermark 45
3.15.1 How to create a text watermark and insert it into the first page 45
3.15.2 How to create an image watermark and insert it into the first page 46
3.15.3 How to remove all watermarks from a page 47
3.16 Barcode 47
3.16.1 How to generate a barcode bitmap from a string 47
3.17 Security 48
3.17.1 How to encrypt a PDF file with Certificate 48
3.17.2 How to encrypt a PDF file with Foxit DRM 49
3.18 Reflow 50
3.18.1 How to create a reflow page and render it to a bmp file 50
3.19 Asynchronous PDF 51
3.20 Pressure Sensitive Ink 52
3.20.1 How to create a PSI and set the related properties for it 52
3.21 Wrapper 52
3.21.1 How to open a document including wrapper data 53
3.22 PDF Objects 53
3.22.1 How to remove some properties from catalog dictionary 53

3.23 Page Object 54
3.23.1 How to create a text object in a PDF page 54
3.23.2 How to add an image logo to a PDF page 55
3.24 Marked content 56
3.24.1 How to get marked content in a page and get the tag name 56
3.25 Layer 57
3.25.1 How to create a PDF layer 57
3.25.2 How to set all the layer nodes information 58
3.25.3 How to edit layer tree 58
3.26 Signature 59
3.26.1 How to sign the PDF document with a signature 60
3.26.2 How to implement signature callback function of signing 61
3.27 Long term validation (LTV) 67
3.27.1 How to establish long term validation of signatures using the default signature callback for subfilter "ETSI.RFC3161" and the default RevocationCallback 68
3.28 PAdES 70
3.29 PDF Action 71
3.29.1 How to create a URI action and insert to a link annot 71
3.29.2 How to create a GoTo action and insert to a link annot 72
3.30 JavaScript 72
3.30.1 How to add JavaScript Action to Document 73
3.30.2 How to add JavaScript Action to Annotation 73
3.30.3 How to add JavaScript Action to FormField 73
3.30.4 How to add a new annotation to PDF using JavaScript 75
3.30.5 How to get/set properties of annotations (strokeColor, fillColor, readOnly, rect, type) using JavaScript 75
3.30.6 How to destroy annotation using JavaScript 76
3.31 Redaction 76
3.31.1 How to redact the text "PDF" on the first page of a PDF 77

3.32 Comparison 79
3.32.1 How to compare two PDF documents and save the differences between them into a PDF file 79
3.33 OCR 81
3.33.1 System requirements 81
3.33.2 Trial limit for SDK OCR add-on module 82
3.33.3 OCR resource files 82
3.33.4 How to run the OCR demo 82
3.34 Compliance 85
3.34.1 System requirements 86
3.34.2 Compliance resource files 86
3.34.3 How to run the compliance demo 86
3.35 Optimization 89
3.35.1 How to optimize PDF files by compressing the color/grayscale/monochrome images 89
3.36 HTML to PDF Conversion 90
3.36.1 System requirements 91
3.36.2 HTML to PDF engine files 91
3.36.3 How to run the html2pdf demo 91
3.36.4 How to work with Html2PDF API 95
3.37 Office to PDF Conversion 96
3.37.1 System requirements 96
3.37.2 How to convert Word to PDF 96
3.37.3 How to convert Excel to PDF 97
3.37.4 How to convert PowerPoint to PDF 97
3.38 Output Preview 98
3.38.1 System requirements 98
3.38.2 How to run the output preview demo 98
3.38.3 How to do output preview using Foxit PDF SDK 98
3.39 Combination 99

3.39.1 How to combine several PDF files into one PDF file 99
3.40 PDF Portfolio 100
3.40.1 System requirements 100
3.40.2 How to create a new and blank PDF portfolio 100
3.40.3 How to create a Portfolio object from a PDF portfolio 101
3.40.4 How to get portfolio nodes 102
3.40.5 How to add file node or folder node 103
3.40.6 How to remove a node 104
3.41 Table Maker 105
3.41.1 System requirements 105
3.41.2 How to add table to a PDF document 105
3.42 Accessibility 107
3.42.1 System requirements 107
3.42.2 How to tag a PDF document 107
FAQ 109
Appendix 112
Supported JavaScript List 112
References 118
Support 119

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 Foxit PDF SDK is your choice

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 they 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**

Do 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 trial version to evaluate 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 the developers who need to integrate Foxit PDF SDK with 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 XP, 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.

2.2 What is in the package

Download Foxit PDF SDK zip for Windows (C API) package and extract it to a new directory "foxitpdfsdk_8_4_2_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 8.4.2, so it shows 8_4_2 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

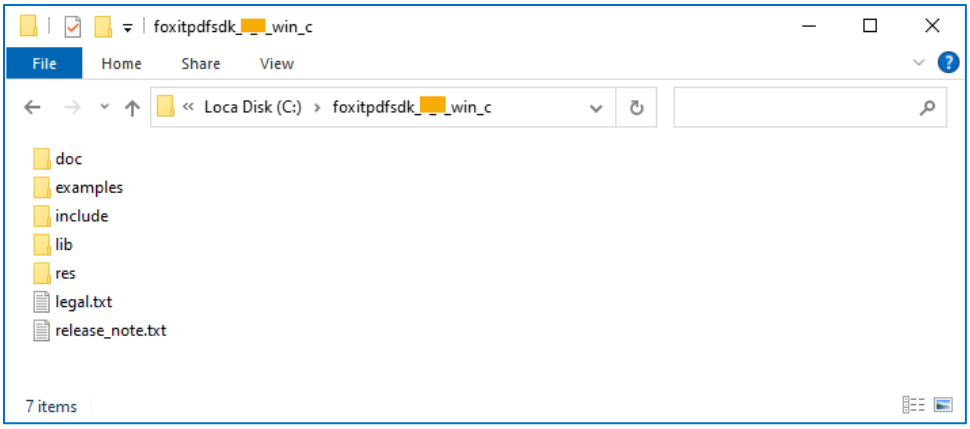


Figure 2-1

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, html2pdf, office2pdf and output preview 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" (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 "*.vxcproj" 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. And the names of the executable files depend on the build configurations.

3) Run a specific executable file, just double-click 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 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 demos

For **ocr** and **compliance** 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 3.33 "OCR" and section 3.34 "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 3.36 "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 3.38 "Output Preview".

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_8_4_2_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.cpp.

```
#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"
```

5) Add using namespace statements.

```
using namespace std;
```

6) 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
```

7) 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=").*

8) 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);
```

9) 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 = static_cast<int>(floatwidth);
```

```
int height = static_cast<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);
```

10) 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.

11) 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.

12) 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.cpp").

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.cpp" 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"

using namespace std;

// 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

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 = static_cast<int>(floatwidth);
int height = static_cast<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 = new char[lFileSize];
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);
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 wstring wstr) {
  std::string curLocale = setlocale(LC_ALL, NULL);
  setlocale(LC_ALL, "chs");
  size_t _Dsize = 2 * wstr.size() + 1;
```

```
 char *_Dest = new char[_Dsize];
 memset(_Dest, 0, _Dsize);
 wcstombs(_Dest, wstr.c_str(), _Dsize);
 std::string result = _Dest;
 delete[] _Dest;
 setlocale(LC_ALL, curLocale.c_str());
 return result;
}

class CFSFile_Read : public FSReaderCallback
{
public:
	CFSFile_Read():m_fileFP(NULL)
		,m_bLargeFile(FALSE)
	{
		user_data = this;
		GetSize = gGetSize;
		ReadBlock = gReadBlock;
		Release = gRelease;
		ifx_streamread = NULL;
	}

	~CFSFile_Read() {}

	bool LoadFile(const wchar_t* wFilePath, bool bLargeFile = false)
	{

		fopen_s(&m_fileFP, wstring2string(wFilePath).c_str(), "rb");
		if (!m_fileFP) return false;

		m_bLargeFile = bLargeFile;
		return true;
	}

	bool LoadFile(const char* filePath, bool bLargeFile = false)
	{
		fopen_s(&m_fileFP, filePath.c_str(), "rb");
		if (!m_fileFP) return false;

		m_bLargeFile = bLargeFile;
		return true;
	}

	static FS_INT64 gGetSize(void* user_data)
	{
		if (((CFSFile_Read*)user_data)->m_bLargeFile)
		{
			_fseeki64(((CFSFile_Read*)user_data)->m_fileFP, 0, SEEK_END);
			long long sizeL = _ftelli64(((CFSFile_Read*)user_data)->m_fileFP);
			return sizeL;
		}
		else
```

```
		{
			fseek(((CFSFile_Read*)user_data)->m_fileFP, 0, SEEK_END);
			return (FS_INT64)ftell(((CFSFile_Read*)user_data)->m_fileFP);
		}
	}

	static FS_BOOL gReadBlock(void* user_data, void* buffer, FS_INT64 offset, size_t size)
	{
		if (((CFSFile_Read*)user_data)->m_bLargeFile)
		{
			_fseeki64(((CFSFile_Read*)user_data)->m_fileFP, offset, SEEK_SET);
			long long readSize = fread(buffer, 1, size, ((CFSFile_Read*)user_data)->m_fileFP);
			return readSize == size ? true : false;
		}
		else
		{
			if (!((CFSFile_Read*)user_data)->m_fileFP) return false;
			if(0 != fseek(((CFSFile_Read*)user_data)->m_fileFP, offset, 0))
				return false;
			if(0 == fread(buffer, size, 1, ((CFSFile_Read*)user_data)->m_fileFP))
				return false;
			return true;
		}
	}

	static void gRelease(void* user_data)
	{
		if(((CFSFile_Read*)user_data)->m_fileFP)
			fclose(((CFSFile_Read*)user_data)->m_fileFP);
		((CFSFile_Read*)user_data)->m_fileFP = NULL;
		delete ((CFSFile_Read*)user_data);
	}

private:
	FILE* m_fileFP;
	bool m_bLargeFile;
}
...
string inputPDFPath = "Sample.pdf";
CFSFile_Read* pFileRead = new CFSFile_Read();
If(!pFileRead->LoadFile(inputPDFPath.c_str())) return;
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.
class  FileWriter:public FSWriterCallback
{
public:
        FileWriter() {
                user_data = this;
                GetSize = gGetSize;
                Release = gRelease;
                Flush = gFlush;
                WriteBlock = gWriteBlock;
        }

        ~FileWriter()  {}

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

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

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

        static FS_BOOL  gReadBlock(void* user_data, void* buffer,FS_INT64 offset,size_t size)    {
                FX_LPBYTE byte_buffer = ((FileWriter*)user_data)->binary_buffer_->GetBuffer();
                memcpy(buffer, byte_buffer + offset, size);
        }

        static void gRelease() {
        }

        CFX_BinaryBuf GetBuffer() {
                return binary_buffer_;
        }
private:
        CFX_BinaryBuf *binary_buffer_;
};
...

FileWriter* filewriter = new FileWriter();

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

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= static_cast<int>(fWidth);
int height= static_cast<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 = static_cast<int>(fWidth);
int height = static_cast<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 progreessive;
FSDK_Renderer_StartRender(renderer, page, matrix, NULL, progreessive);
...
```

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 = static_cast<int>(fWidth);
int height = static_cast<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) {
               wstring exFilePath = "output_directory";
               FS_BOOL bExportStatus;
                FSDK_FileSpec_ExportToFile(file_spec, exFilePath.c_str(), 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"

using namespace std;
...

// 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);
```

```
    string s_text;
    s_text.copy(text_bstr.str, text_bstr.len, 0);
    fwrite(text_bstr.str, sizeof(char), text_bstr.len, file);
    FSDK_WStr_Clear(text);
}
...
```

3.6.2 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);

wstring pattern = L"Foxit";
FS_BOOL return_value3;
FSDK_TextSearch_SetPattern(search, pattern.c_str(), 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 = new FSRectF[return_array_length];
   FSDK_TextSearch_GetMatchRects(search, rect_array, return_array_length);
   FSDK_TextSearch_FindNext(search, return_value5);
   delete[]rect_array;
   }
...
```

3.8 Text Link

In a PDF page, some text contents that represent a hypertext link to a website or a resource on the intent, 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:

3.8.1 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);
...
```

3.9 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:

3.9.1 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);
            }
    }
}
...
```

3.9.2 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"

using namespace std;

wstring string2wstring(const string str) {
  setlocale(LC_ALL, "chs");
  const char* _Source = str.c_str();
  size_t _Dsize = str.size() + 1;
  wchar_t *_Dest = new wchar_t[_Dsize];
  wmemset(_Dest, 0, _Dsize);
  mbstowcs(_Dest, _Source, _Dsize);
```

```
  std::wstring result = _Dest;
  delete[]_Dest;
  setlocale(LC_ALL, "C");
  return result;
}

// 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);
wstring ws_title;
char str[100];
sprintf(str, "A bookmark to a page (index: %d)", i);
ws_title = string2wstring(str);
FS_BOOKMARK_HANDLE child;
FSDK_Bookmark_Insert(root, ws_title.c_str(), e_FSPosLastChild, child);
FSDK_Bookmark_SetDestination(child, dest);
FSDK_Bookmark_SetColor(child, 0xF68C21);
```

3.9.3 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"

using namespace std;

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 = L"";
   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);
}
```

3.10 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:

3.10.1 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);
...
```

3.10.2 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");
}
```

3.10.3 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.
string wstring2string(const wstring wstr) {
  std::string curLocale = setlocale(LC_ALL, NULL);
  setlocale(LC_ALL, "chs");
  size_t _Dsize = 2 * wstr.size() + 1;
  char*_Dest = new char[_Dsize];
  memset(_Dest, 0, _Dsize);
  wcstombs(_Dest, wstr.c_str(), _Dsize);
  std::string result = _Dest;
  delete[]_Dest;
  setlocale(LC_ALL, curLocale.c_str());
  return result;
}
...
FS_FORM_HANDLE form;
FSDK_Form_Create(doc, form);
...
```

```
FS_BOOL return_value;
FSDK_Form_ExportToXML(form, wstring2string(XMLFilePath).c_str(), return_value);
...
```

3.10.4 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;
FSDK_Form_ImportFromXML(form, wstring2string(XMLFilePath).c_str(), return_value);
...
```

3.10.5 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;
FSErrorCode error_code = FSDK_PDFDoc_Create0(input_file.c_str(), 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);

...
```

3.11 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 uses 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:

3.11.1 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"


CFS_XFAAppHandler* pXFAAppHandler = new CFS_XFAAppHandler(); // implement from
foxit::addon::xfa::AppProviderCallback
FSDK_Library_RegisterXFAAppProviderCallback(pXFAAppHandler);
wstring input_file = input_path + L"xfa_dynamic.pdf";
FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file.c_str(), doc);
FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
if (error_code != e_FSErrSuccess) {
        return 1;
}

CFS_XFADocHandler* pXFADocHandler = new CFS_XFADocHandler(); // implement from
foxit::addon::xfa::DocProviderCallback
FS_XFADOC_HANDLE xfa_doc;
FSDK_XFADoc_Create(doc, pXFADocHandler, xfa_doc);
FS_PROGRESSIVE_HANDLE progressive;
FSDK_XFADoc_StartLoad(xfa_doc, NULL, progressive);
...
```

3.11.2 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);
...
```

3.12 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:

3.12.1 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;
form.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);
...
```

3.12.2 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);
    }
}
...
```

3.13 Annotations

3.13.1 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 (TypeWritter/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:

3.13.1.1 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 annnots 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);
...
```

3.13.1.2 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"

wstring string2wstring(const string str) {
  setlocale(LC_ALL, "chs");
  const char* _Source = str.c_str();
  size_t _Dsize = str.size() + 1;
  wchar_t *_Dest = new wchar_t[_Dsize];
  wmemset(_Dest, 0, _Dsize);
  mbstowcs(_Dest, _Source, _Dsize);
  std::wstring result = _Dest;
  delete[]_Dest;
  setlocale(LC_ALL, "C");
  return result;
}
...
// Assuming FS_PDFPAGE_HANDLE page has been loaded and parsed.
// Assuming the annnots 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 = new FSQuadPoints[1];
quad_points_array[0] = quad_points;
FSDK_TextMarkup_SetQuadPoints(highlight, quad_points_array, 1);
delete[] 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, string2wstring(RandomUID()).c_str());
// Appearance should be reset.
FS_BOOL return_code;
FSDK_Annot_ResetAppearanceStream(highlight, return_code);
FSDK_Highlight_Release(highlight);
FSDK_Annot_Release(annot);
...
```

3.13.1.3 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 annnots 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, string2wstring(RandomUID()).c_str());

// 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);
...
```

3.13.1.4 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 = static_cast<int>(fWidth);
int height = static_cast<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);
	...
}
```

3.13.1.5 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"

using namespace std;
...

// 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);
	}
}
```

3.13.1.6 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);
```

3.13.2 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:

3.13.2.1 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"

using namespace std;

string wstring2string(const wstring wstr) {
  std::string curLocale = setlocale(LC_ALL, NULL);
  setlocale(LC_ALL, "chs");
  size_t _Dsize = 2 * wstr.size() + 1;
  char*_Dest = new char[_Dsize];
  memset(_Dest, 0, _Dsize);
  wcstombs(_Dest, wstr.c_str(), _Dsize);
  std::string result = _Dest;
  delete[]_Dest;
  setlocale(LC_ALL, curLocale.c_str());
  return result;
```

```
}
...
// Assuming FS_PDFDOC_HANDLE doc has been loaded.
...
FILE* file = NULL;
#if defined(_WIN32) || defined(_WIN64)
    fopen_s(&file, (const char*)(const char*)wstring2string(fdf_file).c_str(), "rb+");
#else
    file = fopen((const char*)(const char*)wstring2string(fdf_file).c_str(), "rb+");
#endif
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);
```

3.14 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:

3.14.1 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 = static_cast<int>(fWidth);
   int height = static_cast<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);
}
...
```

3.14.2 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.c_str(), 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.c_str(), e_FSSaveFlagNoOriginal, isSave);
...
```

3.15 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:

3.15.1 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
...
```

3.15.2 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.c_str(), 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.
...
```

3.15.3 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
...
```

3.16 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	EAN8	UPCA	EAN13	ITF	PDF417	QR
Dimension	1D	1D	1D	1D	1D	1D	2D	2D

Example:

3.16.1 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.
wstring 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.
wstring 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.c_str(), code_format, unit_width, unit_height,
sz_qr_level, bitmap);
...
```

3.17 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:

3.17.1 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"

...
string wstring2string(const wstring wstr) {
  std::string curLocale = setlocale(LC_ALL, NULL);
  setlocale(LC_ALL, "chs");
```

```
  size_t _Dsize = 2 * wstr.size() + 1;
  char*_Dest = new char[_Dsize];
  memset(_Dest, 0, _Dsize);
  wcstombs(_Dest, wstr.c_str(), _Dsize);
  std::string result = _Dest;
  delete[]_Dest;
  setlocale(LC_ALL, curLocale.c_str());
  return result;
}
...

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file.c_str(), 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 = new FS_BSTR[envelopeslength];
FS_BSTR initial_key;
wstring cert_file_path = input_path + wstring(L"foxit.cer");
if (!GetCertificateInfo((const char*)wstring2string(cert_file_path).c_str(), 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);
delete[]envelopes;
FSDK_PDFDoc_SetSecurityHandler(doc, handler, result);
wstring output_file = output_directory + L"certificate_encrypt.pdf";
FSDK_PDFDoc_SaveAs(doc, output_file.c_str(), e_FSSaveFlagNoOriginal, result);
...
```

3.17.2 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.c_str(), 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";
string 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 = 0xfffffffc;
FS_BOOL result;
FSDK_DRMSecurityHandler_Initialize(handler, encrypt_data, file_id, initialize_key.c_str(), result);
FSDK_PDFDoc_SetSecurityHandler(doc, handler, result);

wstring output_file = output_directory + L"foxit_drm_encrypt.pdf";
FSDK_PDFDoc_SaveAs(doc, output_file.c_str(), e_FSSaveFlagNoOriginal, result);
FSDK_PDFDoc_Release(doc);
...
```

3.18 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:

3.18.1 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_PROGRESSIVE_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_PROGRESSIVE_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_PROGRESSIVE_HANDLE progressive2;
FSDK_Renderer_StartRenderReflowPage(renderer, reflow_page, matrix, NULL, progressive2);
...
```

3.19 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.

3.20 Pressure Sensitive Ink

Pressure **S**ensitive **I**nk (**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:

3.20.1 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);
...
```

3.21 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:

3.21.1 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.c_str(), doc);
FSErrorCode code = 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(wstring2string(file_name).c_str());
...
```

3.22 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 3.23) 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:

3.22.1 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);
}
...
```

3.23 Page Object

Page object is a feature that allows novice users having limited knowledge of PDF objects (see 3.22 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:

3.23.1 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);
...
```

3.23.2 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.c_str(), 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 = static_cast<float>(iwidth);
float height = static_cast<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);
...
```

3.24 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:

3.24.1 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);
}
...
```

3.25 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:

3.25.1 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 ;
}
...
```

3.25.2 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);
    wstring new_name = wstring(L"[View_OFF_Print_ON_Export_Undefined]") + return_GetName.str;
    FSDK_LayerNode_SetName(layer_node, new_name.c_str(), 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);
...
```

3.25.3 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.c_str(), 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);
```

3.26 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:

3.26.1 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";
        SignatureCallbackImpl* sig_callback = new SignatureCallbackImpl(sub_filter);
        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 = AddSiganture(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;
FSDK_Signature_GetState(new_signature, sig_state);
printf("[Before signing] Signed?:%s\t State:%s\r\n",
         is_signed? "true" : "false",
         TransformSignatureStateToString(sig_state).c_str());

// Sign the new signature.
wstring signed_pdf_path = output_directory + L"signed_newsignature.pdf";
if (use_default)
        signed_pdf_path = output_directory + L"signed_newsignature_default_handler.pdf";

wstring cert_file_path = input_path + wstring(L"foxit_all.pfx");
wstring 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.c_str(), cert_file_password.c_str(), e_FSDigestSHA1,
signed_pdf_path.c_str(), 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).c_str());

// Open the signed document and verify the newly added signature (which is the last one).
printf("Signed PDF file: %s\r\n", wstring2string(signed_pdf_path).c_str);
FS_PDFDOC_HANDLE signed_pdf_doc;
FSDK_PDFDoc_Create0(signed_pdf_path.c_str(), 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).c_str());
```

3.26.2 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
class SignatureCallbackImpl : public FSSignatureCallback {
public:
        SignatureCallbackImpl(string subfilter)
                : sub_filter_(subfilter)
                , digest_context_(NULL) {
          user_data = this;
          Release = gRelease;
          StartCalcDigest = gStartCalcDigest;
          ContinueCalcDigest = gContinueCalcDigest;
          GetDigest = gGetDigest;
          Sign = gSign1;
```

```
            Sign0 = gSign;
            VerifySigState = gVerifySigState;
            IsNeedPadData = gIsNeedPadData;
            CheckCertificateValidity = gCheckCertificateValidity;
        }
        ~SignatureCallbackImpl();

        static void gRelease(void* user_data) {
                delete user_data;
        }
        static 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);
        static FSState gContinueCalcDigest(void* user_data, const void* client_data, const FSPauseCallback*
pause);
        static FS_BSTR gGetDigest(void* user_data, const void* client_data);
        static FS_BSTR gSign1(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);
        static FS_BSTR gSign(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);
        static 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);
        static FS_BOOL gIsNeedPadData(void* user_data) {return false;}
        static 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;
        }
protected:
        bool GetTextFromFile(unsigned char *plainString);

        unsigned char* PKCS7Sign(const wchar_t* cert_file_path, string cert_file_password,
                string plain_text, int& signed_data_size);
        bool PKCS7VerifySignature(string signed_data, string plain_text);
        bool ParseP12File(const wchar_t* cert_file_path, string cert_file_password,
                EVP_PKEY** pkey, X509** x509, STACK_OF(X509)** ca);
        ASN1_INTEGER* CreateNonce(int bits);

private:
        string sub_filter_;
        DigestContext* digest_context_;

        string cert_file_path_;
        string cert_file_password_;
};

SignatureCallbackImpl::~SignatureCallbackImpl() {
```

```
	if (digest_context_) {
		delete digest_context_;
		digest_context_ = NULL;
	}
}

bool SignatureCallbackImpl::GetTextFromFile(unsigned char* file_buffer) {
	if (!digest_context_ || !digest_context_->GetFileReadCallback()) return false;
	FSReaderCallback* file_read = digest_context_->GetFileReadCallback();
	file_read->ReadBlock(file_read->user_data, file_buffer, digest_context_->GetByteRangeElement(0),
digest_context_->GetByteRangeElement(1));
	file_read->ReadBlock(file_read->user_data, file_buffer + (digest_context_->GetByteRangeElement(1)-
digest_context_->GetByteRangeElement(0)),
		digest_context_->GetByteRangeElement(2), digest_context_->GetByteRangeElement(3));
	return true;
}

FS_BOOL SignatureCallbackImpl::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 (((SignatureCallbackImpl*) user_data)->digest_context_) {
			delete ((SignatureCallbackImpl*) user_data)->digest_context_;
			((SignatureCallbackImpl*) user_data)->digest_context_ = NULL;
		}
		((SignatureCallbackImpl*) user_data)->digest_context_ = new
DigestContext(const_cast<FSReaderCallback*>(file), byte_range_array, size_of_array);
		if(!SHA1_Init(&((SignatureCallbackImpl*) user_data)->digest_context_->sha_ctx_)) {
			delete ((SignatureCallbackImpl*) user_data)->digest_context_;
			((SignatureCallbackImpl*) user_data)->digest_context_ = NULL;
			return false;
		}
		return true;
}

FSState SignatureCallbackImpl::gContinueCalcDigest(void* user_data, const void* client_data, const
FSPauseCallback* pause) {
	if (!((SignatureCallbackImpl*)user_data)->digest_context_) return e_FSError;

	FS_UINT32 file_length =
((SignatureCallbackImpl*)user_data)->digest_context_->GetByteRangeElement(1) +
((SignatureCallbackImpl*)user_data)->digest_context_->GetByteRangeElement(3);
	unsigned char* file_buffer = (unsigned char*)malloc(file_length);
	if (!file_buffer || !((SignatureCallbackImpl*)user_data)->GetTextFromFile(file_buffer)) return e_FSError;

	SHA1_Update(&((SignatureCallbackImpl*)user_data)->digest_context_->sha_ctx_, file_buffer, file_length);
	free(file_buffer);
	return e_FSFinished;
}

FS_BSTR SignatureCallbackImpl::gGetDigest(void* user_data, const void* client_data) {
	 FS_BSTR digest;
	if (!((SignatureCallbackImpl*)user_data)->digest_context_){
```

```
            digest.str = "";
            digest.len = 0;
            return digest;
        }
        unsigned char* md = reinterpret_cast<unsigned char*>(OPENSSL_malloc((SHA_DIGEST_LENGTH)*
sizeof(unsigned char)));
        if (1 != SHA1_Final(md, &((SignatureCallbackImpl*)user_data)->digest_context_->sha_ctx_)){
        digest.str = "";
            digest.len = 0;
            return digest;
        }
        FSDK_BStr_SetLength(digest, SHA_DIGEST_LENGTH);
        FSDK_BStr_Set(digest, (char*)(reinterpret_cast<const char*>(md)), SHA_DIGEST_LENGTH);
        OPENSSL_free(md);
        return digest;
}

FS_BSTR SignatureCallbackImpl::gSign1(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;
                bstr.str = "";
                bstr.len = 0;
                if (!((SignatureCallbackImpl*)user_data)->digest_context_) return bstr;
                string plain_text;
                if ("adbe.pkcs7.sha1" == ((SignatureCallbackImpl*)user_data)->sub_filter_) {
                        plain_text = string((const char*)digest, digest_length);
                }
                int signed_data_length = 0;
                unsigned char* signed_data_buffer =
((SignatureCallbackImpl*)user_data)->PKCS7Sign(cert_path,
        wstring2string(password), plain_text, signed_data_length);
                if (!signed_data_buffer) return bstr;
                FS_BSTR signed_data;
                FSDK_BStr_SetLength(signed_data, signed_data_length);
                FSDK_BStr_Set(signed_data, (char*)(const char*)signed_data_buffer, signed_data_length);
                free(signed_data_buffer);
                return signed_data;
}

FS_BSTR SignatureCallbackImpl::gSign(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 SignatureCallbackImpl::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.
                    if (!((SignatureCallbackImpl*)user_data)->digest_context_) return e_FSStateVerifyErrorData;
                    string plain_text;
                    unsigned char* file_buffer = NULL;
                    if ("adbe.pkcs7.sha1" == ((SignatureCallbackImpl*)user_data)->sub_filter_) {
                            plain_text = string(reinterpret_cast<const char*>(digest), digest_length);
                    } else {
                            return e_FSStatesStateUnknown;
                    }

                    string signed_data_str = string(reinterpret_cast<const char*>(signed_data), signed_data_len);
                    bool ret = ((SignatureCallbackImpl*)user_data)->PKCS7VerifySignature(signed_data_str,
plain_text);
                    if (file_buffer) free(file_buffer);
                    return ret ? e_FSStateVerifyNoChange : e_FSStateVerifyChange;

}

ASN1_INTEGER* SignatureCallbackImpl::CreateNonce(int bits) {
            unsigned char buf[20];
            int len = (bits - 1) / 8 + 1;
            // Generating random byte sequence.
            if (len > (int)sizeof(buf)) {
                    return NULL;
            }
            if (RAND_bytes(buf, len) <= 0) {
                    return NULL;
            }
            // Find the first non-zero byte and creating ASN1_INTEGER object.
            int i = 0;
            for (i = 0; i < len && !buf[i]; ++i) ;
            ASN1_INTEGER* nonce = NULL;
            if (!(nonce = ASN1_INTEGER_new())) {
                    ASN1_INTEGER_free(nonce);
                    return NULL;
            }
            OPENSSL_free(nonce->data);
            // Allocate at least one byte.
            nonce->length = len - i;
            if (!(nonce->data = reinterpret_cast<unsigned char*>(OPENSSL_malloc(nonce->length + 1)))) {
                    ASN1_INTEGER_free(nonce);
                    return NULL;
            }
            memcpy(nonce->data, buf + i, nonce->length);
            return nonce;
}
```

```
bool SignatureCallbackImpl::ParseP12File(const wchar_t* cert_file_path, String cert_file_password,
        EVP_PKEY** pkey, X509** x509, STACK_OF(X509)** ca) {
                FILE* file = NULL;
#if defined(_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* pkcs12 = d2i_PKCS12_fp(file, NULL);
                fclose (file);
                if (!pkcs12) {
                        return false;
                }

                if (!PKCS12_parse(pkcs12, (const char*)cert_file_password, pkey, x509, ca)) {
                        return false;
                }

                PKCS12_free(pkcs12);
                if (!pkey)
                        return false;
                return true;
}

unsigned char* SignatureCallbackImpl::PKCS7Sign(const wchar_t* cert_file_path, String cert_file_password,
        String plain_text, int& signed_data_size) {
                PKCS7* p7 = NULL;
                EVP_PKEY* pkey = NULL;
                X509* x509 = NULL;
                STACK_OF(X509)* ca = NULL;
                if(!ParseP12File(cert_file_path, cert_file_password, &pkey, &x509, &ca))
                        return NULL;

                p7 = PKCS7_new();
                PKCS7_set_type(p7, NID_pkcs7_signed);
                PKCS7_content_new(p7, NID_pkcs7_data);

                // Application should not judge the sign algorithm with the content's length.
                // Here, just for convenient;
                if (plain_text.GetLength() > 32)
                        PKCS7_ctrl(p7, PKCS7_OP_SET_DETACHED_SIGNATURE, 1, NULL);

                PKCS7_SIGNER_INFO* signer_info = PKCS7_add_signature(p7, x509, pkey, EVP_sha1());
                PKCS7_add_certificate(p7, x509);

                for (int i = 0; i< sk_num(CHECKED_STACK_OF(X509,ca)); i++)
                        PKCS7_add_certificate(p7, (X509*)sk_value(CHECKED_STACK_OF(X509,ca), i));
```

```
                // Set source data to BIO.
                BIO* p7bio = PKCS7_dataInit(p7, NULL);
                BIO_write(p7bio, plain_text.GetBuffer(1), plain_text.GetLength());
                PKCS7_dataFinal(p7, p7bio);

                FREE_CERT_KEY;
                BIO_free_all(p7bio);
                // Get signed data.
                unsigned long der_length = i2d_PKCS7(p7, NULL);
                unsigned char* der = reinterpret_cast<unsigned char*>(malloc(der_length));
                memset(der, 0, der_length);
                unsigned char* der_temp = der;
                i2d_PKCS7(p7, &der_temp);
                PKCS7_free(p7);
                signed_data_size = der_length;
                return (unsigned char*)der;
}

bool SignatureCallbackImpl::PKCS7VerifySignature(String signed_data, String plain_text) {
        // Retain PKCS7 object from signed data.
        BIO* vin = BIO_new_mem_buf((void*)signed_data.GetBuffer(1), signed_data.GetLength());
        PKCS7* p7 = d2i_PKCS7_bio(vin, NULL);
        STACK_OF(PKCS7_SIGNER_INFO) *sk = PKCS7_get_signer_info(p7);
        int sign_count = sk_PKCS7_SIGNER_INFO_num(sk);

        int length = 0;
        bool bSigAppr = false;
        unsigned char *p = NULL;
        for(int i=0;i<sign_count; i++) {
                PKCS7_SIGNER_INFO* sign_info = sk_PKCS7_SIGNER_INFO_value(sk,i);

                BIO *p7bio = BIO_new_mem_buf((void*)plain_text.GetBuffer(1), plain_text.GetLength());
                X509 *x509= PKCS7_cert_from_signer_info(p7,sign_info);
                if(1 == PKCS7_verify(p7, NULL, NULL,p7bio, NULL, PKCS7_NOVERIFY))
                        bSigAppr = true;
                BIO_free(p7bio);
        }
        PKCS7_free(p7);
        BIO_free(vin);
        return bSigAppr;
}
```

3.27 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:

3.27.1 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_TIMESTAMPSERVER_HANDLE timestamp_server;
FS_WSTR password;
password.str = L"";
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.c_str(),pdf_doc);
FS_PROGRESSIVE_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= new
FS_SIGNATUREVERIFYRESULT_HANDLE[length];
		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];
			FSLTVState 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]);
		}
		delete[] 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_PROGRESSIVE_HANDLE sign_progressive;
FSDK_Signature_StartSign(timestamp_signature, L"", L"", e_FSDigestSHA256, saved_ltv_pdf_path.c_str(), 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.c_str(), check_pdf_doc);
FS_PROGRESSIVE_HANDLE progressive1;
FSDK_PDFDoc_StartLoad(check_pdf_doc, NULL, true, NULL, progressive1);
{
		// Use LTVVeirfier 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=newFS_SIGNATUREVERIFYRESULT_HAN
DLE[length];
        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.
        }
        delete[] sig_verify_result_array;
}

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

3.28 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.

3.29 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:

3.29.1 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 annnots 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, recf, 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);
```

3.29.2 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, recf, 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);
```

3.30 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:

3.30.1 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, javascipt_action);
FSDK_JavaScriptAction_SetScript(javascipt_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, javascipt_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerDocWillClose, result);
...
```

3.30.2 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, javascipt_action);
FSDK_JavaScriptAction_SetScript(javascipt_action, L"app.alert(\"Hello Foxit \");");
FS_ADDITIONALACTION_HANDLE additional_act;
FSDK_AdditionalAction_Create2(annot, additional_act);
FSDK_AdditionalAction_SetAction(additional_act, e_FSTriggerAnnotMouseButtonPressed, javascipt_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotMouseButtonPressed, result)
...
```

3.30.3 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 javascipt_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, javascipt_action);
FSDK_JavaScriptAction_SetScript(javascipt_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, javascipt_action);
// Update text field's appearance.
FS_WIDGETANNOT_HANDLE widget2;
FSDK_Control_GetWidget(control2, widget2);
FS_BOOL result2;
FSDK_Annot_ResetAppearanceStream(widget2, result2);
```

```
...
```

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

3.30.5 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 javascipt_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, javascipt_action);
FSDK_JavaScriptAction_SetScript(javascipt_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, javascipt_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 javascipt_action1 ;
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, javascipt_action);
```

```
FSDK_JavaScriptAction_SetScript(javascipt_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, javascipt_action);
FS_BOOL result1;
FSDK_AdditionalAction_DoJSAction(additional_act1, e_FSTriggerAnnotCursorEnter, result1);
...
```

3.30.6 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 javascipt_action;
FS_PDFDOC_HANDLE doc;
FSDK_Form_GetDocument(form, doc);
(FS_JAVASCRIPTACTION_HANDLE)FSDK_Action_Create(doc, e_FSTypeJavaScript, javascipt_action);
FSDK_JavaScriptAction_SetScript(javascipt_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, javascipt_action);
FS_BOOL result;
FSDK_AdditionalAction_DoJSAction(additional_act, e_FSTriggerAnnotCursorEnter, result);
...
```

3.31 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_FSErrInvalidLicense.

- 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:

3.31.1 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_PROGRESSIVE_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);
std::vector<FSRectF> rect_array;
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 = new FSRectF[return_array_length];
            FSDK_TextSearch_GetMatchRects(text_search, rect, return_array_length);
            for (int i = 0; i < return_array_length; i++) {
              FSRectF rectF = rect[i];
              rect_array.push_back(rectF);
            }
            delete[] rect;
```

```
            FSDK_TextSearch_FindNext(text_search, return_FindNext);
 }
if(rect_array.GetSize() > 0) {
            FSRectF* rect = new FSRectF[rectarray.size()];
            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);
            delete[] rect;
            FS_BOOL return_code;
            FSDK_Annot_ResetAppearanceStream(redact, return_code);
            wstring save_pdf_patth = output_directory + wstring(L"AboutFoxit_redected_default.pdf");
            FS_PROGRESSIVE_HANDLE return_StartSaveAs1;
            FSDK_PDFDoc_StartSaveAs(doc, save_pdf_patth.c_str(), 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);
            wstring save_pdf_patth1 = output_directory + wstring(L"AboutFoxit_redected_setColor.pdf");
            FS_PROGRESSIVE_HANDLE return_StartSaveAs2;
            FSDK_PDFDoc_StartSaveAs(doc, save_pdf_patth1.c_str(), e_FSSaveFlagsSaveFlagNormal, NULL,
return_StartSaveAs2);

            FSDK_Markup_SetOpacity(redact, (float)0.5);
            FSDK_Annot_ResetAppearanceStream(redact, return_code);
            wstring save_pdf_patth2 = output_directory + wstring(L"AboutFoxit_redected_setOpacity.pdf");
            FS_PROGRESSIVE_HANDLE return_StartSaveAs3;
            FSDK_PDFDoc_StartSaveAs(doc, save_pdf_patth2.c_str(), e_FSSaveFlagsSaveFlagNormal, NULL,
return_StartSaveAs3);
            FSDK_Redaction_Apply(redaction, return_code);
            if(return_code)
                        cout << "Redact page(0) succeed." << endl;
            else
                        cout << "Redact page(0) failed." << endl;
 }
wstring save_pdf_patth3 = output_directory + wstring(L"AboutFoxit_redected_apply.pdf");

FS_PROGRESSIVE_HANDLE return_StartSaveAs;

FSDK_PDFDoc_StartSaveAs(doc, save_pdf_patth3.c_str(), e_FSSaveFlagsSaveFlagNormal, NULL,
return_StartSaveAs);
```

3.32 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:

3.32.1 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.c_str(), base_doc);
FSErrorCode error_code = 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.c_str(), 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.
FSCompareResults 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 = new 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 = new 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[i].rect_array = new
FSRectF[result.base_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 =new
FSRectF[result.compared_doc_results_array[i].array_length_rect_array];
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)
   {
      char res_string_new[100];
      wstring diffcontents = item.diff_contents.str;
      sprintf_s(res_string_new, "\"%s\"", wstring2string(diffcontents).c_str());
      string res_string = res_string_new;

      // Add stamp to mark the "delete" type differences between the two documents.
      CreateDeleteTextStamp(page, item.rect_array, item.array_length_rect_array, 0xff0000,
string2wstring(res_string).c_str(), L"Compare : Delete", L"Text");
   }
   else if (type == e_FSCompareResultTypeInsertText)
   {
      char res_string_new[100];
      wstring diffcontents = item.diff_contents.str;
      sprintf_s(res_string_new, "\"%s\"", wstring2string(diffcontents).c_str());
      string res_string = res_string_new;

      // Highlight the "insert" type differences between the two documents.
      CreateDeleteText(page, item.rect_array, item.array_length_rect_array, 0x0000ff,
string2wstring(res_string).c_str(), L"Compare : Insert", L"Text");
   }
   else if (type == e_FSCompareResultTypeReplaceText)
   {
      char res_string_new[100];
      wstring diffcontents = item.diff_contents.str;
      wstring diffcontents2 = result.base_doc_results_array[i].diff_contents.str;
```

```
        sprintf_s(res_string_new, "[Old]: \"%s\"\r\n[New]: \"%s\"", wstring2string(diffcontents2).c_str(),
wstring2string(diffcontents).c_str());
        string res_string = res_string_new;

        // Squiggly the "replace" type differences between the two documents.
        CreateSquigglyRect(page, item.rect_array, item.array_length_rect_array, 0xe7651a,
string2wstring(res_string).c_str(), L"Compare : Replace", L"Text");
    }
}

// Save the comparison result to a PDF file.
FS_BOOL return_result = false;

wstring output_compared_doc = output_directory + wstring(L"new.pdf");

FSDK_PDFDoc_SaveAs(compared_doc, output_compared_doc.c_str(), 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.*

3.33 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 8.4, the OCR engine has been upgraded, please contact Foxit support team or sales team to get the latest engine files package.

From version 8.4.2, the OCR engine has also 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).

3.33.1 System requirements

Platform: Windows

Programming Language: C, C++, Java, Python, C#

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

SDK Version: Foxit PDF SDK (C++, Java, C#) 6.4 or higher; Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher

3.33.2 Trial limit for SDK OCR add-on module

For the trial version, there are three trail 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.

3.33.3 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.

3.33.4 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.

3.33.4.1 Load the OCR demo in Visual Studio

To load the OCR demo in your specific environment, please choose one of the following ways:

1) Load the visual studio solution files "simple_demo_vs2010.sln" or "simple_demo_vs2015.sln" or "simple_demo_vs2017.sln" (depending on your Visual Studio version) in the "\examples\simple_demo" folder. Right-click the ocr demo, choose **Set as StartUp Project**.

2) Load the "ocr_vs2010.vcxproj" or "ocr_vs2015.vcxproj" or "ocr_vs2017.vcxproj" (depending on your Visual Studio version) in the "\examples\simple_demo\ocr" folder.

3.33.4.2 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.

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.*

3.33.4.3 Configure the demo

After building the OCR resource directory, configure the demo in the "\examples\simple_demo\ocr**ocr.cpp**" 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");
```

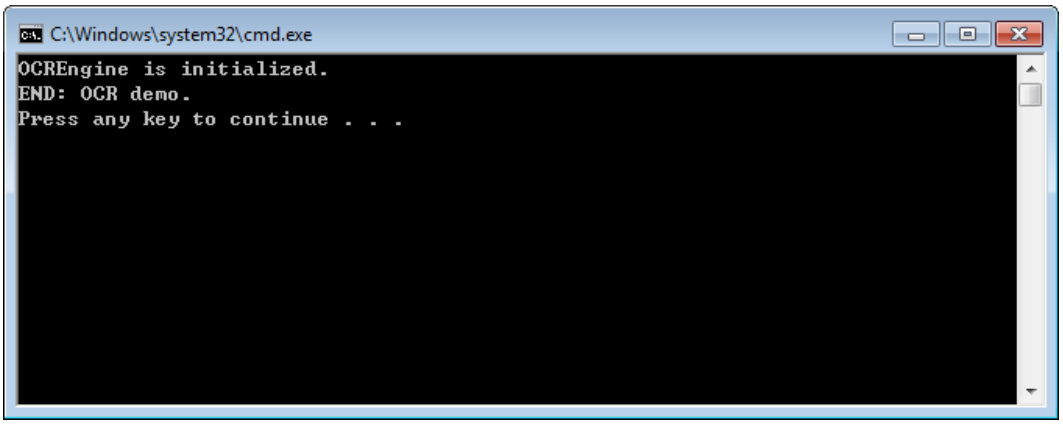
(Optional) Set log for OCREngine

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)
FSDK_OCREngine_SetLogFile((output_directory+L"ocr.log").c_str());
```

3.33.4.4 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

3.34 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).

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

3.34.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C

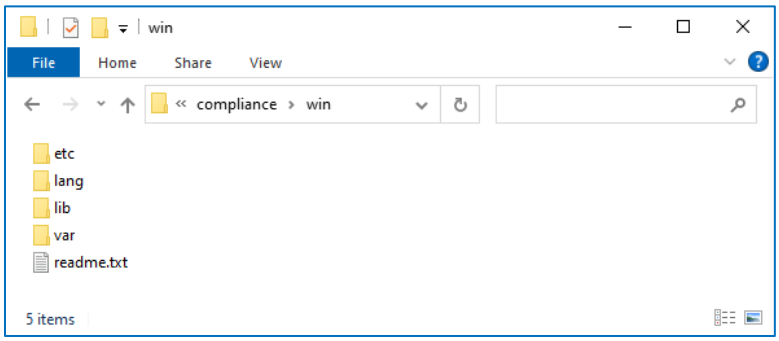
License Key requirement: 'Compliance' module permission in the license key

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

3.34.2 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:



3.34.3 How to run the compliance demo

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.

3.34.3.1 Build a compliance resource directory

Before running the **compliance** 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.

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

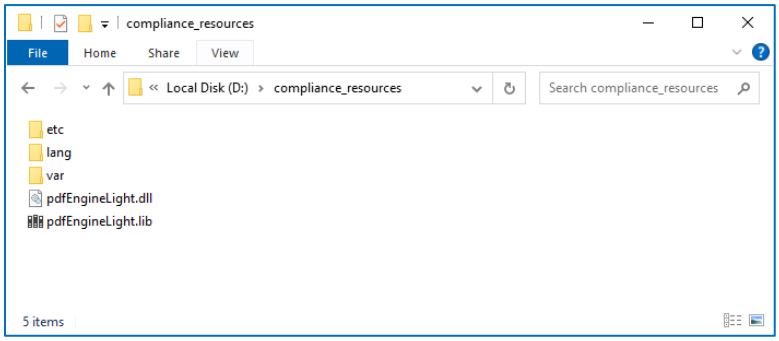
1) Create a new folder to add the resources. For example, "D:/compliance_resources".

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:



3.34.3.2 Configure the demo

After building the compliance resource directory, configure the demo in the "\examples\simple_demo\compliance**compliance.cpp**" file.

To load the **compliance** demo in Visual Studio, please choose one of the following ways:

1) Load the visual studio solution files "simple_demo_vs2010.sln" or "simple_demo_vs2015.sln" or "simple_demo_vs2017.sln" (depending on your Visual Studio version) in the "\examples\simple_demo" folder. Right-click the compliance demo, choose **Set as StartUp Project**.

2) Load the "compliance_vs2010.vcxproj" or "compliance_vs2015.vcxproj" or "compliance_vs2017.vcxproj" (depending on your Visual Studio version) in the "\examples\simple_demo\compliance" folder.

Specify the compliance resource directory

In the "compliance.cpp" 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.
  wstring 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 = "";

  if (compliance_resource_folder_path.empty()) {
    std::cout << "compliance_resource_folder_path is still empty. Please set it with a valid path to compliance resource folder path." << std::endl;
    return 1;
  }
  // Initialize compliance engine.
  FSErrorCode error_code = FSDK_ComplianceEngine_Initialize(compliance_resource_folder_path.c_str(), compliance_engine_unlockcode);
```

***Note**: If you have purchased an authorization license key (includes 'Compliance' module permission), Foxit sales team will send you an extra unlock code for initializing compliance engine.*

(Optional) Set language for compliance engine

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

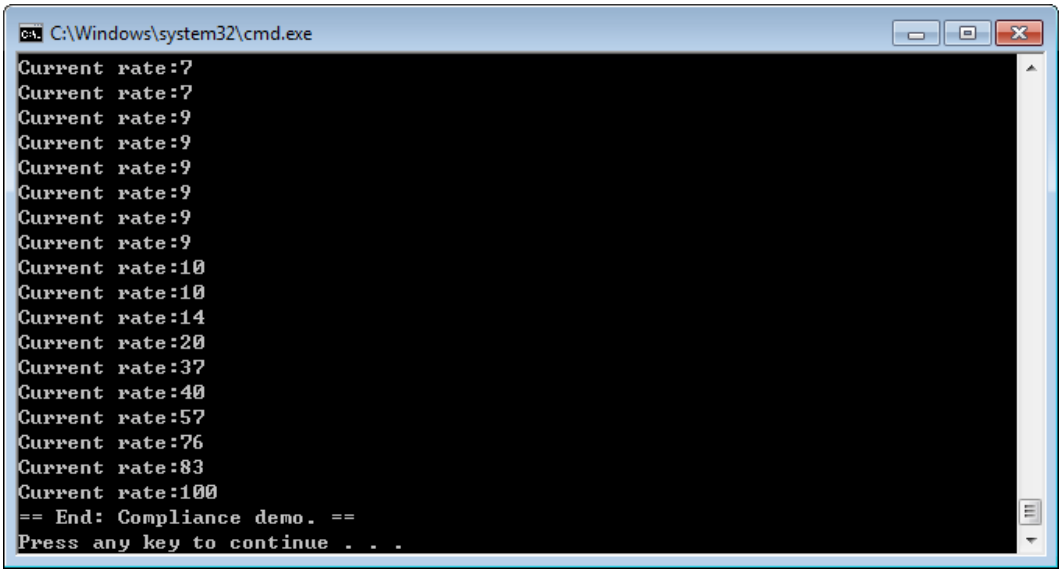
FSDK_ComplianceEngine_SetTempFolderPath function is used to set a temp folder to store several files for proper processing (e.g verifying or converting). If no custom temp folder is set by this function, the default temp folder in system will be used.

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");
```

3.34.3.3 Run the demo

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



The demo will

- verify whether the PDF ("\examples\simple_demo\input_files\AboutFoxit.pdf") is compliance with PDF/A-1a standard, and convert the PDF to be compliance with PDF/A-1a standard.
- convert PDF file ("\examples\simple_demo\input_files\AF_ImageXObject_FormXObject.pdf") to PDF-1.4 and PDF-1.7.

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

3.35 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 date, 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:

3.35.1 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"
```

```
using namespace std;

FS_PDFDOC_HANDLE doc;
FSDK_PDFDoc_Create0(input_file.c_str(), 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);
cout << "Optimized Start." << endl;
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, "Optimize progress percent: %d %", percent);
        std::cout<<res_string<<std::endl;
}
if(e_FSFinished ==  progress_state)
{
        wstring output_directory_Optimized = output_directory + L"ImageCompression_Optimized.pdf";
        FS_BOOL return_result = false;
        FSDK_PDFDoc_SaveAs(doc, output_directory_Optimized.c_str(),
e_FSSaveFlagRemoveRedundantObjects, return_result);
}
cout << "Optimized Finish." << endl;
```

3.36 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.

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.

From version 8.2, the HTML to PDF engine has been upgraded, please contact Foxit support team or sales team to get the latest engine files package.

From version 8.3, the HTML to PDF engine has also been upgraded, please contact Foxit support team or sales team to get the latest engine files package.

From version 8.4.1, the HTML to PDF engine has also 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 running the 'html2pdf' demo.

3.36.1 System requirements

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

Programming Language: C, C++, Java, C#, Python, Objective-C

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

3.36.2 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: "**D:/htmltopdf/win**").

3.36.3 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.

3.36.3.1 Configure the demo

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

To load the html2pdf demo in Visual Studio, please choose one of the following ways:

1) Load the visual studio solution files "simple_demo_vs2010.sln" or "simple_demo_vs2015.sln" or "simple_demo_vs2017.sln" (depending on your Visual Studio version) in the

"\examples\simple_demo" folder. Right-click the html2pdf demo, choose **Set as StartUp Project**.

2) Load the "html2pdf_vs2010.vcxproj" or "html2pdf_vs2015.vcxproj" or "html2pdf_vs2017.vcxproj" (depending on your Visual Studio version) in the "\examples\simple_demo\html2pdf" folder.

Specify the html2pdf engine directory

In the "html2pdf.cpp" 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 converting html to pdf. Please refer to Developer Guide for more details.
wstring engine_path = L"D:/htmltopdf/win/fxhtml2pdf.exe";  // or  engine_path = L"D:/htmltopdf/win/fxhtml2pdf";
```

(Optional) Specify cookies file path

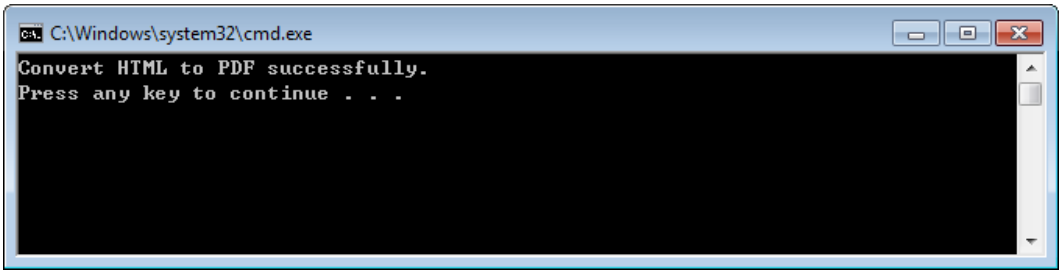
Add the path of the cookies file exported from the web pages that you want to convert. For example,

```
// "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.
wstring cookies_path = L"D:/cookies.txt";
```

3.36.3.2 Run the demo

Run the demo without parameters

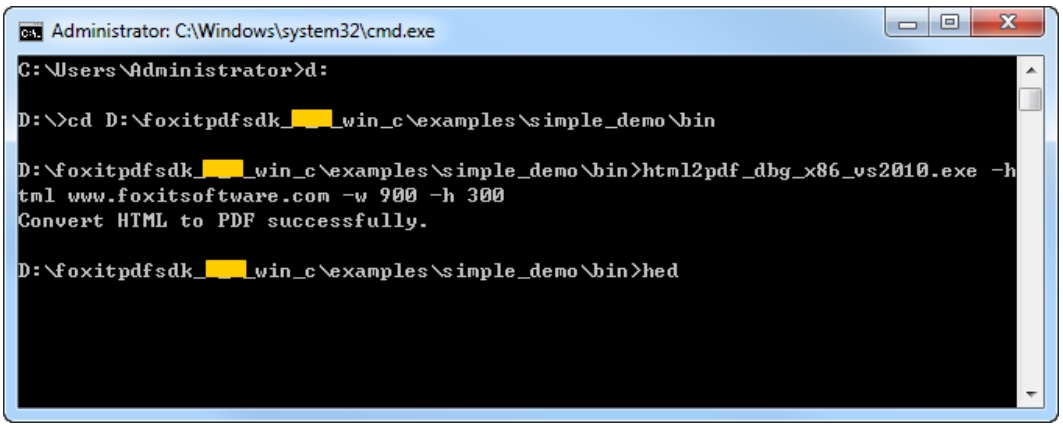
Once you run the demo successfully, the console will print the following by default:



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:



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.

Parameters	Description
-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. • 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.
-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. • 0 : Single page mode. • 1 : Multiple pages mode.
-scale	The scaling mode. • 0 : No need to scale pages. • 1 : Scale pages. • 2 : Enlarge page.
-link	Whether to convert links. • 'yes' : Convert links. • 'no' : No need to convert links.
-tag	Whether to generate tag. • 'yes' : Generate tag. • 'no' : No need to generate tag.
-bookmarks	Whether to generate bookmarks. • 'yes' : Generate bookmarks . • 'no' : No need to generate bookmarks.
-print_background	Whether to print background. • 'yes' : Print bookmarks . • 'no' : No need to print bookmarks.
-optimize_tag	Whether to optimize tag tree. • 'yes' : Optimize tag tree . • 'no' : No need to optimize tag tree.

Parameters	Description
-media	The media style. • 0 : Screen media style. • 1 : Print media style.
-encoding	The HTML encoding format. • 0 : Auto encoding . • 1-73 : Other encodings.
-render_images	Whether to render images. • 'yes' : Render images. • 'no' : No need to render images.
-remove_underline_for_link	Whether to remove underline for link. • 'yes' : Remove underline for link. • 'no' : No need to remove underline for link.
-headerfooter	Whether to generate headerfooter. • '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. • '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.

3.36.4 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 Office to PDF Conversion

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 converst 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 printerk is already set on your Windows system.

3.37.1 System requirements

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

Programming Language: C, C++, Python, Java, C#

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)

Example:

3.37.2 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.

wstring word_file_path = L"test.doc";
wstring 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.c_str(), L"", output_path.c_str(), word_convert_setting_data);
```

3.37.3 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.

wstring excel_file_path = L"test.xls";
wstring 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.c_str(), L"", output_path.c_str(), excel_convert_setting_data);
```

3.37.4 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.

wstring ppt_file_path = L"test.ppt";
wstring 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.c_str(), L"", output_path.c_str(), ppt_convert_setting_data);
```

3.38 Output Preview

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

3.38.1 System requirements

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

Programming Language: C, C++, Java, C#, Python, Objective-C

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

3.38.2 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.
wstring 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.

3.38.3 How to do output preview using Foxit PDF SDK

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

#include <string>
using namespace std;

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

// Set folder path which contains default icc profile files.
FSErrorCode 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);
wstring simulation_icc_file_path = L"icc_profile.icc";
error_code = FSDK_OutputPreview_SetSimulationProfile(output_preview, simulation_icc_file_path.c_str());
FSDK_OutputPreview_SetShowType(output_preview, e_FSShowAll);
```

```
FS_UINT32 process_plates_length = 0;
FSDK_OutputPreview_GetPlates(output_preview, e_FSColorantTypeProcess, NULL, process_plates_length);
FS_BSTR* process_plates = new 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);
FS_BSTR *spot_plates = new 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].str, true);
}
delete[] 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].str, true);
}
delete[] spot_plates;

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

3.39 Combination

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

3.39.1 How to combine several PDF files into one PDF file

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

#include <string>
using namespace std;

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

int info_array_size = 3;
FS_COMBINEDOCUMENTINFO_HANDLE* info_array = new
FS_COMBINEDOCUMENTINFO_HANDLE[info_array_size];
FS_WSTR password;
password.str = L"";
password.len = 0;
auto item = info_array[0];
FSDK_CombineDocumentInfo_Create((input_path + wstring(L"AboutFoxit.pdf")).c_str(), &password,
info_array[0]);
```

```
item = info_array[0];
FSDK_CombineDocumentInfo_Create((input_path + wstring(L"Annot_all.pdf")).c_str(), &password, info_array[1]);
item = info_array[1];
FSDK_CombineDocumentInfo_Create((input_path + wstring(L"SamplePDF.pdf")).c_str(), &password,
info_array[2]);
item = info_array[2];

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

FS_PROGRESSIVE_HANDLE progressive = NULL;
FSDK_Combination_StartCombineDocuments((output_directory + L"Test_Combined.pdf").c_str(), 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]);
}
delete[] info_array;
```

3.40 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.

3.40.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C

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

Example:

3.40.2 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;
}
```

3.40.3 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.c_str(), 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;
}
```

3.40.4 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 = new FS_PORTFOLIONODE_HANDLE[array_length];
FSDK_PortfolioFolderNode_GetSortedSubNodes(root_folder, sub_node_array, array_length);
for (FS_UINT32index = 0; index < array_length; index++) {
    FS_PORTFOLIONODE_HANDLE node = sub_node_array[index];
    FSPortfolioNodeType node_type = e_FSPortfolioNodeTypeTypeUnknown;
    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 = new 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]);
            }
            delete[] 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]);
    }
    delete[] 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;
    }
```

3.40.5 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.
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.
MyStreamCallback* my_stream_callback = new MyStreamCallback();
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;
}
```

3.40.6 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);
```

3.41 Table Maker

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

3.41.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C

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

SDK Version: Foxit PDF SDK 8.4 or higher

3.41.2 How to add table to a PDF document

Foxit PDF SDK provides an electronictable 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;
wstring 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);
    cell_array[row][col].cell_text.str = (wchar_t*)cell_text[index].c_str();
    cell_array[row][col].cell_text.len = cell_text[index].length();
    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);
}
```

3.42 Accessibility

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

3.42.1 System requirements

Platform: Windows, Linux, Mac

Programming Language: C, C++, Java, C#, Python, Objective-C

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

SDK Version: Foxit PDF SDK 8.4 or higher

3.42.2 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.c_str(), 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.c_str(), 0, return_value);
FSDK_TaggedPDF_Release(tagged_pdf);
FSDK_PDFDoc_Release(pdf_doc);
...
```

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"

using namespace std;
...

bool ChangeTextObjectContent()
{
    wstring input_file = input_path + L"AboutFoxit.pdf";
    FS_PDFDOC_HANDLE doc;
    FSDK_PDFDoc_Create0(input_file.c_str(), doc);
    FSErrorCode error_code = FSDK_PDFDoc_Load(doc, NULL);
    if (error_code != e_FSErrSuccess) {
            FSDK_PDFDoc_Release(doc);
            printf("The Doc [%s] Error: %d\n", (const char*)wstring2string(input_file).c_str(), 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 = new 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);
    wstring output_directory = output_path + wstring(L"graphics_objects/");
    wstring output_file = output_directory + L"After_revise.pdf";
    FSDK_PDFDoc_SaveAs(doc, output_file.c_str(), 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 module on Windows 7 when running the OCR simple demo, even if the engine file has been upgraded to the latest and the engine path in the demo has been configured 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".

APPENDIX

Supported JavaScript List

Objects' property or method

Object	Properites/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
annotation method	destroy	V7.0
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
app methods	alert	V4.0
	beep	V4.0
	browseForDoc	V4.0
	clearInterval	V4.0
	clearTimeOut	V4.0

Object	Properites/Method Names	Minimum Supported SDK Version
	launchURL	V4.0
	mailMsg	V4.0
	response	V4.0
	setInterval	V4.0
	setTimeOut	V4.0
	popUpMenu	V4.0
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
	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

Object	Properites/Method Names	Minimum Supported SDK Version
	path	V4.0
	producer	V4.0
	subject	V4.0
	title	V4.0
document methods	addAnnot	V7.0
	addField	V4.0
	addIcon	V4.0
	calculateNow	V4.0
	createDataObject	V6.2
	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
event properties	change	V4.0
	changeEx	V4.0
	commitKey	V4.0
	fieldFull	V4.0
	keyDown	V4.0

Object	Properites/Method Names	Minimum Supported SDK Version
	modifier	V4.0
	name	V4.0
	rc	V4.0
	selEnd	V4.0
	selStart	V4.0
	shift	V4.0
	source	V4.0
	target	V4.0
	targetName	V4.0
	type	V4.0
	value	V4.0
	willCommit	V4.0
event methods	-	-
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

Object	Properites/Method Names	Minimum Supported SDK Version
	multiline	V4.0
	multipleSelection	V4.0
	name	V4.0
	numItems	V4.0
	page	V4.0
	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
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

Object	Properites/Method Names	Minimum Supported SDK Version
global methods	setPersistent	V4.0
Icon properties	name	V4.0
util methods	printd	V4.0
	printf	V4.0
	printx	V4.0
	scand	V4.0
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

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
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 Where has lots of helpful articles that might help with solving your issue.

Phone Support:

Phone: 1-866-MYFOXIT or 1-866-693-6948