



DEVELOPER GUIDE FOXIT PDF SDK

For JAVA

TABLE OF CONTENTS

1	Introduction to Foxit PDF SDK	1
1.1	Why Choose Foxit PDF SDK.....	1
1.2	Foxit PDF SDK for Java.....	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 initialize Foxit PDF SDK.....	4
2.4	How to run a demo	5
2.4.1	Run a demo using batch script.....	6
2.4.2	Run a demo using Eclipse IDE.....	7
2.5	How to create a simple project.....	8
3	Working with SDK API.....	15
3.1	Initialize Library	15
3.1.1	How to initialize Foxit PDF SDK.....	15
3.2	Document.....	16
3.2.1	How to create a PDF document from scratch.....	16
3.2.2	How to load an existing PDF document from file path	16
3.2.3	How to load an existing PDF document from a memory buffer	17
3.2.4	How to load an existing PDF document from a file read callback object.....	17
3.2.5	How to load PDF document and get the first page of the PDF document.....	18
3.2.6	How to save a PDF to a file.....	18
3.2.7	How to save a document into memory buffer by FileWriterCallback	19
3.3	Page.....	19

3.3.1	How to get page size	20
3.3.2	How to calculate bounding box of page contents.....	20
3.3.3	How to create a PDF page and set the size	20
3.3.4	How to delete a PDF page	20
3.3.5	How to flatten a PDF page.....	21
3.3.6	How to get and set page thumbnails in a PDF document.....	21
3.4	Render.....	21
3.4.1	How to render a page to a bitmap	22
3.4.2	How to render page and annotation	23
3.5	Attachment.....	23
3.5.1	How to export the embedded attachment file from a PDF and save it as a single file	23
3.5.2	How to remove all the attachments of a PDF	24
3.6	Text Page	24
3.6.1	How to extract text from a PDF page.....	25
3.6.2	How to select text of a rectangle area in a PDF.....	25
3.7	Text Search.....	25
3.7.1	How to search a text pattern in a PDF.....	25
3.8	Search and Replace	26
3.8.1	System requirements	26
3.8.2	How to work with the search and replace function	26
3.9	Text Link.....	27
3.9.1	How to retrieve hyperlinks in a PDF page	27
3.10	Bookmark	27
3.10.1	How to find and list all bookmarks of a PDF.....	28
3.10.2	How to insert a new bookmark	29
3.10.3	How to create a table of contents based on bookmark information in PDFs	29
3.11	Form (AcroForm)	29
3.11.1	How to load the forms in a PDF.....	30
3.11.2	How to count form fields and get the properties.....	30
3.11.3	How to export the form data in a PDF to a XML file	31

3.11.4	How to import form data from a XML file	31
3.11.5	How to get coordinates of a form field.....	31
3.12	XFA Form.....	32
3.12.1	How to load XFADoc and represent an Interactive XFA form	32
3.12.2	How to export and import XFA form data.....	33
3.13	Form Design	34
3.13.1	How to add a text form field to a PDF	34
3.13.2	How to remove a text form field from a PDF.....	34
3.14	Annotations	35
3.14.1	General	35
3.14.2	Import annotations from or export annotations to a FDF file.....	39
3.15	Image Conversion.....	40
3.15.1	How to convert PDF pages to bitmap files	40
3.15.2	How to convert a image file to PDF file.....	42
3.16	Watermark.....	42
3.16.1	How to create a text watermark and insert it into the first page	42
3.16.2	How to create an image watermark and insert it into the first page.....	43
3.16.3	How to remove all watermarks from a page	44
3.17	Barcode.....	44
3.17.1	How to generate a barcode bitmap from a string	45
3.18	Security	45
3.18.1	How to encrypt a PDF file with Certificate.....	45
3.18.2	How to encrypt a PDF file with Foxit DRM.....	48
3.19	Reflow	49
3.19.1	How to create a reflow page and render it to a bmp file.	49
3.20	Asynchronous PDF	51
3.21	Pressure Sensitive Ink.....	51
3.21.1	How to create a PSI and set the related properties for it.....	51
3.22	Wrapper	52

3.22.1	How to open a document including wrapper data	52
3.23	PDF Objects	53
3.23.1	How to remove some properties from catalog dictionary	53
3.24	Page Object	53
3.24.1	How to create a text object in a PDF page	53
3.24.2	How to add an image logo to a PDF page	54
3.25	Marked content	55
3.25.1	How to get marked content in a page and get the tag name	55
3.26	Layer	55
3.26.1	How to create a PDF layer	56
3.26.2	How to set all the layer nodes information	56
3.26.3	How to edit layer tree	56
3.27	Signature	57
3.27.1	How to sign the PDF document with a signature	57
3.27.2	How to implement signature callback function of signing	58
3.28	Long term validation (LTV)	60
3.28.1	How to establish long term validation of signatures using the default signature callback for subfilter "ETSI.RFC3161" and the default RevocationCallback	60
3.29	PADES	61
3.30	PDF Action	62
3.30.1	How to create a URI action and insert to a link annot	62
3.30.2	How to create a GoTo action and insert to a link annot	63
3.31	JavaScript	63
3.31.1	How to add JavaScript Action to Document	63
3.31.2	How to add JavaScript Action to Annotation	64
3.31.3	How to add JavaScript Action to FormField	64
3.31.4	How to add a new annotation to PDF using JavaScript	65
3.31.5	How to get/set properties of annotations (strokeColor, fillColor, readOnly, rect, type) using JavaScript	65
3.31.6	How to destroy annotation using JavaScript	66

3.32	Redaction	66
3.32.1	How to redact the text "PDF" on the first page of a PDF	67
3.33	Comparison	68
3.33.1	How to compare two PDF documents and save the differences between them into a PDF file	68
3.34	OCR	69
3.34.1	System requirements	70
3.34.2	Trial limit for SDK OCR add-on module	70
3.34.3	OCR resource files	70
3.34.4	How to run the OCR demo	71
3.35	Compliance	74
3.35.1	System requirements	74
3.35.2	Compliance resource files	75
3.35.3	How to run the compliance demo	76
3.36	Optimization	80
3.36.1	How to optimize PDF files by compressing the color/grayscale/monochrome images	80
3.37	HTML to PDF Conversion	81
3.37.1	System requirements	81
3.37.2	HTML to PDF engine files	81
3.37.3	How to run the html2pdf demo	82
3.37.4	How to work with Html2PDF API	86
3.37.5	How to get HTML data from stream and convert it to a PDF file	86
3.38	Office to PDF Conversion	88
3.38.1	System requirements	88
3.38.2	How to convert Word to PDF	89
3.38.3	How to convert Excel to PDF	89
3.38.4	How to convert PowerPoint to PDF	90
3.39	Output Preview	91
3.39.1	System requirements	91
3.39.2	How to run the output preview demo	91

3.39.3	How to do output preview using Foxit PDF SDK.....	91
3.40	Combination.....	92
3.40.1	How to combine several PDF files into one PDF file	92
3.41	PDF Portfolio	93
3.41.1	System requirements	93
3.41.2	How to create a new and blank PDF portfolio.....	93
3.41.3	How to create a Portfolio object from a PDF portfolio	93
3.41.4	How to get portfolio nodes	94
3.41.5	How to add file node or folder node	95
3.41.6	How to remove a node.....	95
3.42	Table Maker.....	96
3.42.1	System requirements	96
3.42.2	How to add table to a PDF document.....	96
3.43	Accessibility	97
3.43.1	System requirements	97
3.43.2	How to tag a PDF document	98
3.44	PDF to Office Conversion	98
3.44.1	System requirements	98
3.44.2	PDF to Office resource files.....	98
3.44.3	How to run the pdf2office demo	100
3.44.4	How to work with PDF2office API.....	101
3.45	DWG to PDF Conversion.....	102
3.45.1	System requirements	102
3.45.2	DWG To PDF engine files.....	102
3.45.3	How to run the dwg2pdf demo	102
3.45.4	How to convert DWG to PDF.....	103
3.46	OFD.....	103
3.46.1	System requirements	104
3.46.2	OFD engine file	104
3.46.3	How to run the ofd demo.....	104

3.46.4	How to implement the conversion between OFD file and PDF file.....	104
3.46.5	How to render OFD page	105
3.47	Paragraph Editing.....	106
3.47.1	System requirements	107
3.47.2	How to work with paragraph editing.....	107
3.48	3D Rendering	110
3.48.1	System requirements	110
3.48.2	How to display the 3D annotation	110
3.48.3	How to set render mode and controller.....	111
FAQ		112
Appendix		116
	Supported JavaScript List.....	116
References		130
Support.....		131

1 INTRODUCTION TO FOXIT PDF SDK

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

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

1.1 Why Choose Foxit PDF SDK

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

- **Easy to integrate**

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

- **Lightweight footprint**

Does not exhaust system resource and deploys quickly.

- **Cross-platform support**

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

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

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

- **Premium World-side Support**

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

1.2 Foxit PDF SDK for Java

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 Java API on Windows, Linux and Mac platforms.

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

1.3 Evaluation

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

1.4 License

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

1.5 About this guide

This guide is intended for developers who need to integrate Foxit PDF SDK for Java into their own applications. It aims at introducing installation package structure on desktop platform with Java, and the usage of SDK.

2 GETTING STARTED

It is 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

Platform	System Requirement	JDK version	Note
Windows	Windows Vista, 7, 8 and 10 (32-bit and 64-bit) Windows Server 2003, 2008 and 2012 (32-bit and 64-bit)	At least 1.8	It only supports for Windows 8/10 classic style, but not for Store App or Universal App.
Linux	32-bit and 64-bit OS armv7/armv8 Starting from version 8.4 Foxit PDF SDK for Linux (x86 and x64), the minimum supported version of GCC compiler has been upgraded from gcc4.8 to gcc4.9.4.	At least 1.8	All Linux for x86/x64 samples have been tested on Ubuntu14.0 32/64 bit. All Linux for armv7/armv8 samples have been tested on armv7 or armv8 OS.
Mac	64-bit and Arm64 OS	At least 1.8	

2.2 What is in the Package

Package for Windows Java is named "foxitpdfsdk_10_0_win_java.zip", package for Linux x86/x64 Java is named "foxitpdfsdk_10_0_linux_java.zip", package for Linux armv7/armv8 Java is named "foxitpdfsdk_10_0_linux_arm_java.zip", package for Mac x64 Java is named "foxitpdfsdk_10_0_mac_java.zip" and package for Mac arm64 Java is named "foxitpdfsdk_10_0_mac_arm64_java.zip". They have the similar structure, so in this guide mainly introduce "foxitpdfsdk_10_0_win_java.zip" as an example.

Download the package for Windows Java and extract it to a new directory like "foxitpdfsdk_10_0_win_java". The structure of the release package is shown in Figure 2-1.

NOTE: the highlighted rectangle in the figure is just the version of Foxit PDF SDK. Here the SDK version is 10.0, so it shows 10_0 in the package name. Other highlighted rectangles in other figures have the same meaning in this guide.

This package contains the following folders:

- doc:** API references, developer guide
- examples:** sample projects and demos
- lib:** libraries and license files
- res:** the default icc profile files used for output preview demo

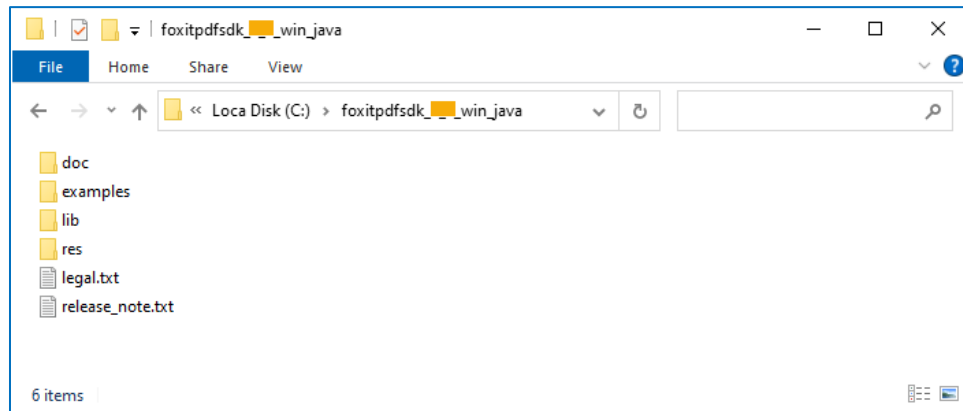


Figure 2-1

Note: For Mac arm64 Java package, it doesn't have the **res** directory. Therefore, it doesn't support the **output preview** demo.

2.3 How to initialize Foxit PDF SDK

It is necessary for applications to initialize Foxit PDF SDK with license before calling any APIs. The static function **Library.initialize**(sn, key) is provided in com.foxit.sdk.common.Library. The trial license files can be found in the "lib" folder. Before initializing library, you should load the proper library for your system at first.

The following code will automatically load the proper library and then initialize the library:

```
// Load the library.  
// You can also use System.load("filename") instead. The filename argument must be an absolute path name.  
static {  
    String os = System.getProperty("os.name").toLowerCase();  
    String lib = "fsdk_java_";  
    if (os.startsWith("win")) {  
        lib += "win";  
    } else if (os.startsWith("mac")) {  
        lib += "mac";  
    } else {  
        lib += "linux";  
    }  
}
```

```

if (System.getProperty("sun.arch.data.model").equals("64")) {
    if(System.getProperty("os.arch").equals("aarch64")){
        if(os.startsWith("mac")){
            lib += "arm";
        }
        else{
            lib += "armv8";
        }
    }
    else{
        lib += "64";
    }
} else {
    if(System.getProperty("os.arch").equals("arm")){
        lib += "armv7";
    }
    else{
        lib += "32";
    }
}
System.loadLibrary(lib);
}

// Initialize the library.
// 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=").
String sn = " ";
String key = " ";
int error_code = Library.initialize(sn, key);
if (error_code != e_ErrSuccess)
    return;
...

```

2.4 How to run a demo

GCC compiler update on Linux

Starting from version 8.4 Foxit PDF SDK for Linux (x86 and x64), the minimum supported version of GCC compiler has been upgraded from gcc4.8 to gcc4.9.4. For the SDK to work properly, make sure your current GCC version is 4.9.4 or higher, or the libstdc++.so.6 is 6.0.20 or higher.

The following is the list for minimum version support for current release of Foxit PDF SDK:

OS	Tool chain	GLIBC
Linux x86/x64	gcc 4.9.4	GLIBC_2.17
Linux armv7	gcc-arm-8.3-2019.03-x86_64-arm-linux-gnueabihf	GLIBC_2.28
Linux armv8	gcc-arm-8.3-2019.03-x86_64-aarch64-linux-gnu	GLIBC_2.27

clang compiler update on Mac x64

Starting from version 9.0, the version of clang used for building and compiling the Foxit PDF SDK for Mac (x64) has been upgraded from 9.1.0 to 11.0.3.

2.4.1 Run a demo using batch script

Foxit PDF SDK provides several simple demos in directory "`\examples\simple_demo`". All these demos (except `office2pdf`, `pdf2office`, `ocr` and `ofd` demo for Windows and Linux; `html2pdf`, `compliance`, `output preview` and `dwg2dpf` demos for Windows, Linux and Mac) can be run directly with the ".bat" files or ".sh" files in directory "`\examples\simple_demo`":

- a. Go to directory "`\examples\simple_demo`", and run all demos by "`RunAllDemo.bat`" for Windows or by "`RunAllDemo.sh`" for Linux and Mac.
- b. If you want to run a specific single demo, please locate to the directory of the demo, for example locate to "`\examples\simple_demo\annotation`", and run this demo by "`RunDemo.bat`" for Windows or by "`RunDemo.sh`" for Linux and Mac.

"`\examples\simple_demo\input_files`" contains all the input files used among these demos. Some demos will generate output files (pdf, text or image files) to a folder named by the demo name under "`\examples\simple_demo\output_files`".

OCR (support Windows, Linux x64)

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

Compliance demos (support Windows, Linux (x86 and x64), Mac)

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

HTML to PDF demo (support Windows, Linux (x86 and x64), Mac)

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

Office to PDF demo (support Windows, Linux (x86 and x64))

For **office2pdf** demo, you need to refer to section 3.38 "[Office to PDF Conversion](#)".

Output Preview demo (support Windows, Linux (x86 and x64), Mac (x64))

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

PDF to Office demo (support Windows, Linux)

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

Dwg to PDF demo (support Windows, Linux (x86 and x64), Mac(x64))

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

OFD demo (Windows, Linux (x64 and armv8))

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

2.4.2 Run a demo using Eclipse IDE

In order to reduce the size of the package, we did not provide the Eclipse project for the demos. Therefore, if you want to run a demo in Eclipse IDE, you should do the steps below:

- 1) Create a Java project in Eclipse, and then refer to the steps 1)-3) in the next section "[How to create a simple project](#)" to integrate the libraries into the project.
- 2) Copy the java file of the demo that you want to run, and then paste it to the project. For example, for image2pdf demo, copy the "image2pdf.java" file (under "\examples\simple_demo\image2pdf") to the "src" folder of the project.
- 3) Open the Java file, modify the *output_path* and *input_path* path as your need. For example, if you want to use the default test files in the code, you may need to use an absolute path which points to the "output_files" and "input_files" folders in the "examples\simple_demo".
- 4) Run the demo. Right-click the project, choose **Run As -> Java Application** to run the project.

Note:

- For **ocr** and **compliance** demos, you should build a resource directory referring to section 3.34 "[OCR](#)" and section 3.35 "[Compliance](#)".

- For **html2pdf** demo, you should get the engine files package for converting from HTML to PDF, and then specify the html2pdf engine directory, referring to section 3.37 "[HTML to PDF Conversion](#)".
- 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.
- For **output preview** demo, you should set the folder path which contains default icc profile files referring to section 3.39 "[Output Preview](#)".
- For **pdf2office** demo, you should get the engine files package for converting from PDF to office, and then specify the pdf2office engine directory and metrics data, referring to section 3.44 "[PDF to Office Conversion](#)".
- For **dwg2pdf** demo, you should get the engine files package for converting from DWG to PDF, and then specify the dwg2pdf engine directory, referring to section 3.45 "[DWG to PDF Conversion](#)".
- For **ofd** demo, you should get the OFD engine files package, and then specify the ofd engine directory, referring to section 3.46 "[OFD](#)".

2.5 How to create a simple project

In this section, we will show you how to use Foxit PDF SDK for Windows Java (Linux Java and Mac Java are similar to Windows Java) to create a simple project that renders the first page of a PDF to a bitmap and saves it as a JPG image. For better writing code, we use Eclipse IDE to create a Java project called "test". Then follow the steps below:

- 1) Copy "lib" folder from the download package to the project folder, and then refresh the project. The structure of the *test* project is shown in Figure 2-2.

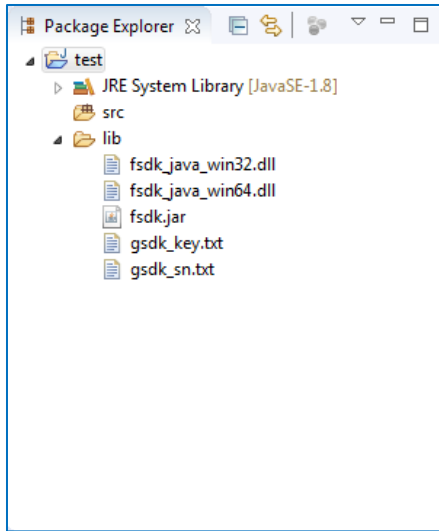


Figure 2-2

- 2) Add "fsdk.jar" to the project. Right click the *test* project, select "Build Path > Configure Build Path > Libraries > Add JARs", and choose the "fsdk.jar" in "test/lib" as shown in Figure 2-3.

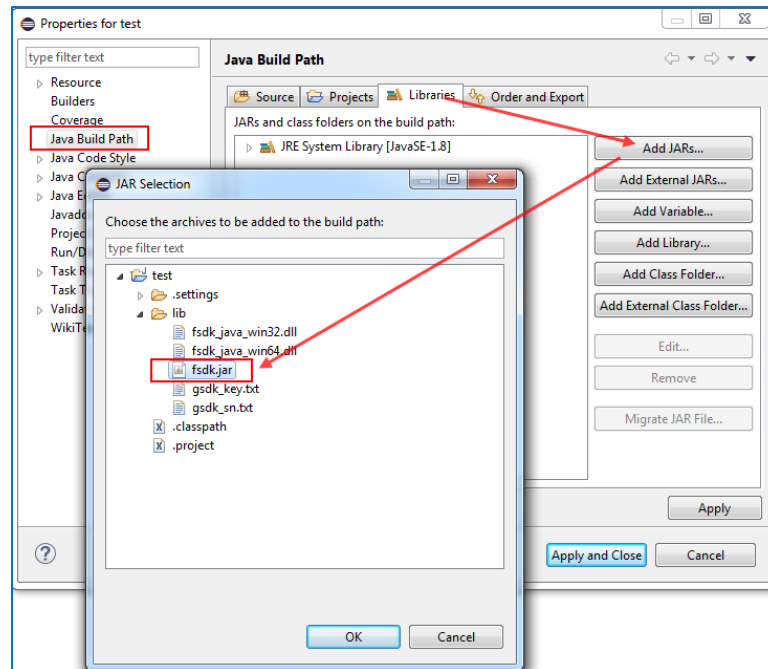


Figure 2-3

- 3) Configure build path for .dll (for Windows) or .so (for Linux) or .dylib (for Mac). Right click the *test* project, select "Build Path > Configure Build Path > Source > Native library location: (None) > Edit", and locate to the "test/lib" folder as shown in Figure 2-4.

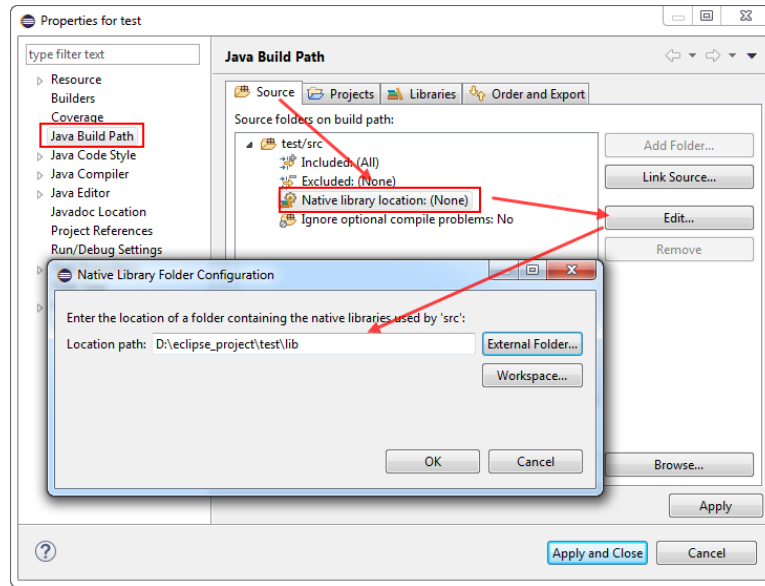


Figure 2-4

- 4) Create a new class file called test.java under "test/src/test" directory.
- 5) Open the "test.java" file, import the classes that you need to use in "fsdk.jar". Here, we just import the classes as follows:

```
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcr.Matrix2D;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
```

- 6) Load and initialize the PDF SDK library. The project can automatically load the proper library for your system with the following code. If your system is 64 bit, the fsdk_java_win64.dll library will be loaded, or loading the fsdk_java_win32.dll library. Then, initialize the library with the license files. The trial license files can be found in the "lib" folder.

```
// Load the library.
static {
    if(System.getProperty("sun.arch.data.model").equals("64")) {
        System.loadLibrary("fsdk_java_win64");
    } else {
        System.loadLibrary("fsdk_java_win32");
    }
}

// Initialize the library.
// 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=").
int error_code = Library.initialize("sn", "key");
if (error_code != e_ErrSuccess) {
    return;
}
```

- 7) Load a PDF document, and parse the first page of the document. Assume that you have already put a "Sample.pdf" to the "test" folder.

```
// load a "Sample.pdf" document.
PDFDoc doc = new PDFDoc("Sample.pdf");
error_code = doc.load(null);
if (error_code != e_ErrSuccess) {
    return;
}

// Get the first page of the document.
PDFPage page = doc.getPage(0);

// Parse page.
page.startParse(e_ParsePageNormal, null, false);
```

- 8) Render the first page to a bitmap and save it as a JPG file.

```
int width = (int) page.getWidth();
int height = (int) page.getHeight();
Matrix2D matrix = page.getDisplayMatrix(0, 0, width, height, page.getRotation());

// Prepare a bitmap for rendering.
Bitmap bitmap = new Bitmap(width, height, e_DIBArgb, null, 0);
bitmap.fillRect(0xFFFFFFFF, null);

// Render page
Rendered render = new Rendered(bitmap, false);
render.startRender(page, matrix, null);

// Add the bitmap to image and save the image.
Image image = new Image();
image.addFrame(bitmap);
image.saveAs("testpage.jpg");
```

- 9) Build and run the project. Right-click the *test* project in package explorer, and then choose "Run as > Java Application" to run it. The "testpage.jpg" will be generated in the "test" folder (See Figure 2-5).

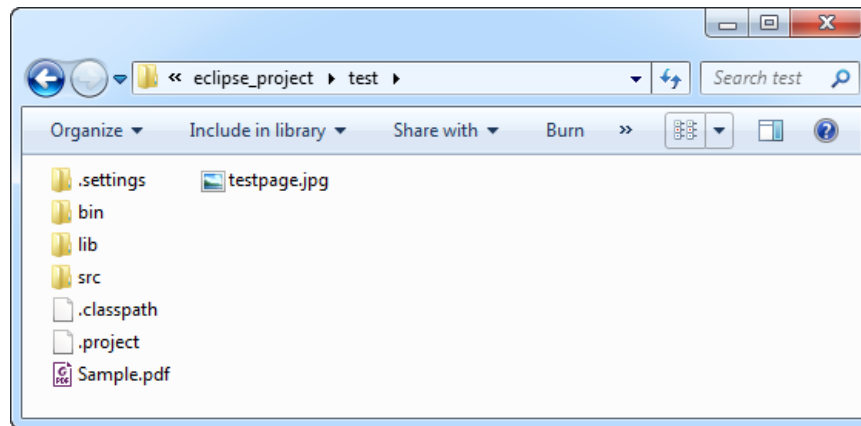


Figure 2-5

The final contents of "test.java" is as follow:

```
package test;

import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcrd.Matrix2D;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;

public class test {

    // Load the library.
    static {
        if(System.getProperty("sun.arch.data.model").equals("64")) {
            System.loadLibrary("fsdk_java_win64");
        } else {
            System.loadLibrary("fsdk_java_win32");
        }
    }

    public static void main(String[] args) throws PDFException {

        // 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=").
String sn = " ";
String key = " ";
int error_code = Library.initialize(sn, key);
if (error_code != e_ErrSuccess) {
    return;
}

// load a "Sample.pdf" document.
PDFDoc doc = new PDFDoc("Sample.pdf");
error_code = doc.load(null);
if (error_code != e_ErrSuccess) {
    return;
}

// Get the first page of the document.
PDFPage page = doc.getPage(0);

// Parse page.
page.startParse(e_ParsePageNormal, null, false);

int width = (int) page.getWidth();
int height = (int) page.getHeight();
Matrix2D matrix = page.getDisplayMatrix(0, 0, width, height, page.getRotation());

// Prepare a bitmap for rendering.
Bitmap bitmap = new Bitmap(width, height, e_DIBArgb, null, 0);
bitmap.fillRect(0xFFFFFFFF, null);

// Render page.
Renderer render = new Renderer(bitmap, false);
render.startRender(page, matrix, null);

// Add the bitmap to image and save the image.
Image image = new Image();
image.addFrame(bitmap);
image.saveAs("testpage.jpg");
}
```

Note: If you have not installed Eclipse IDE in your machine, you can create a Java file, and then use command line to build and run it. For example, we use the test.java file created in the previous Eclipse IDE, and then follow the steps below:

1. Create a folder named "test_cm". Put the test.java into this folder. (Please note that you need to delete the first line "package test;" in the "test.java").
2. Copy "lib" folder to the same folder (test_cm) with "test.java".
3. Put a PDF document named "Sample.pdf" into "test_cm" folder.

4. *Open command line window and navigate to "test_cm" folder, and then use the following commands to build and run the Java file.*

For Windows,

```
javac -cp ./lib/fsdk.jar *.java  
java -Djava.library.path=lib -classpath ./lib/fsdk.jar test
```

For Linux and Mac,

```
javac -cp ../lib/fsdk.jar *.java  
java -Djava.library.path=lib -classpath ./lib/fsdk.jar test
```

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 Java 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 and unlock Foxit PDF SDK license before calling any APIs. The function `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 `Library.release` to release it. Before initializing library, you should load the proper library for your system at first.

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

```
import com.foxit.sdk.common.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;

// Load the library.
// You can also use System.load("filename") instead. The filename argument must be an absolute path name.
static {
    String os = System.getProperty("os.name").toLowerCase();
    String lib = "fsdk_java_";
    if (os.startsWith("win")) {
        lib += "win";
    } else if (os.startsWith("mac")) {
        lib += "mac";
    } else {
        lib += "linux";
    }

    if (System.getProperty("sun.arch.data.model").equals("64")) {
        if (System.getProperty("os.arch").equals("aarch64")) {
            if (os.startsWith("mac")) {
                lib += "arm";
            }
        } else {
            lib += "armv8";
        }
    }
}
```

```
        else{
            lib += "64";
        }
    } else {
        if(System.getProperty("os.arch").equals("arm")){
            lib += "armv7";
        }
        else{
            lib += "32";
        }
    }
    System.loadLibrary(lib);
}

// Initialize the library.
String sn = "";
String key = "";
int error_code = Library.initialize(sn, key);
if (error_code != e_ErrSuccess)
    return;
...
```

3.2 Document

A PDF document object can be constructed with an existing PDF file from file path, memory buffer, a custom implemented ReaderCallback object and an input file stream. Then call function [PDFDoc.Load](#) or [PDFDoc.StartLoad](#) to load document content. A PDF document object 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

```
import static com.foxit.sdk.pdf.PDFDoc.*;
...

PDFDoc doc = new PDFDoc();
```

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

3.2.2 How to load an existing PDF document from file path

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
...

PDFDoc doc = new PDFDoc("sample.pdf");
int error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;
```


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

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import java.io.FileInputStream;
import java.io.BufferedInputStream;
...

BufferedInputStream bis = new BufferedInputStream(new FileInputStream("sample.pdf"));
byte[] b = new byte[bis.available()];
bis.read(b);
PDFDoc doc = new PDFDoc(b);
error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;
```

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

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import java.io.IOException;
import com.foxit.sdk.common.fxcr.FileReaderCallback;
import java.io.RandomAccessFile;
...

class FileReader extends FileReaderCallback {
    private RandomAccessFile file_ = null;
    FileReader() {
    }

    boolean LoadFile(String file_path) throws FileNotFoundException {
        file_ = new RandomAccessFile(file_path, "r");
        return true;
    }

    @Override
    public long getSize() {
        try {
            return this.file_.length();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return 0;
    }

    @Override
    public boolean readBlock(byte[] buffer, long offset, long size) {

        try {
            file_.seek(offset);
            int read = file_.read(buffer, 0, (int) size);
            return read == size ? true : false;
        }
```

```
    } catch (IOException e) {
        e.printStackTrace();
    }
    return false;
}

public void release() {
    try {
        this.file_.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}

FileReader callback = new FileReader();
callback.LoadFile("sample.pdf");
PDFDoc doc = new PDFDoc(callback, false);
int error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;
```

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

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.pdf.PDFPage.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;
...

PDFDoc doc = new PDFDoc("sample.pdf");
int error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;

// Get the first page of the document.
PDFPage page = doc.getPage(0);
// Parse page.
page.startParse(e_ParsePageNormal, null, false);
```

3.2.6 How to save a PDF to a file

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.pdf.PDFPage.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import static com.foxit.sdk.pdf.PDFDoc.e_SaveFlagNoOriginal;
...

PDFDoc doc = new PDFDoc("sample.pdf");
int error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;
```

```
doc.saveAs("new_Sample.pdf", e_SaveFlagNoOriginal);
```

3.2.7 How to save a document into memory buffer by FileWriterCallback

```
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import java.io.FileInputStream;
import java.io.BufferedInputStream;
...

class FileWriter extends FileWriterCallback {
    private ByteArrayOutputStream buf = new ByteArrayOutputStream();

    @Override
    public long getSize() {
        return buf.size();
    }

    @Override
    public boolean writeBlock(byte[] buffer, long offset, long size) {
        buf.write(buffer, (int)0, (int)size);
        return true;
    }

    @Override
    public void release() {
    }

    @Override
    public boolean flush() {
        return true;
    }
}

...
FileWriter filewriter = new FileWriter();

// Assuming PDFDoc doc has been loaded.
...

doc.startSaveAs(filewriter, e_SaveFlagNoOriginal, null);
...
```

3.3 Page

PDF Page is the basic and important component of PDF Document. A [PDFPage](#) object is retrieved from a PDF document by function [PDFDoc.getPage](#). Page level APIs provide functions to parse, render, edit (includes creating, deleting and flattening) a page, retrieve PDF annotations, read and set the properties of a page, and etc. For most cases, A PDF page needs to be parsed before it is rendered or processed.

Example:

3.3.1 How to get page size

```
import static com.foxit.sdk.pdf.PDFPage.*;
...

// Assuming PDFPage page has been loaded and parsed.
...
int width = (int) page.getWidth();
int height = (int) page.getHeight();
...
```

3.3.2 How to calculate bounding box of page contents

```
import static com.foxit.sdk.pdf.PDFPage.*;
...

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

RectF calcRc = page.calcContentBBox(e_CalcContentsBox);
float fcalcRc = calcRc.width();
float fcalcRcHeight = calcRc.height();
```

3.3.3 How to create a PDF page and set the size

```
import static com.foxit.sdk.pdf.PDFPage.*;
...

// Assuming PDFDoc doc has been loaded.

// Insert a new blank PDF page to document, which will be inserted to the first page.
PDFPage newBlankPage = doc.insertPage(-1, 500, 800);

// Insert a new blank PDF page to document, which will be inserted at index 1.
PDFPage newBlankPage = doc.insertPage(1, 500, 800);

// Insert a new blank PDF page to document, which will be inserted to the end.
PDFPage newBlankPage = doc.insertPage(doc.getPageCount(), e_SizeLetter);
```

3.3.4 How to delete a PDF page

```
import static com.foxit.sdk.pdf.PDFDoc.*;
...

// Assuming PDFDoc doc has been loaded.

// Remove a PDF page by page index.
doc.removePage(index);
```

```
// Remove a specified PDF page.  
doc.removePage(page);  
...
```

3.3.5 How to flatten a PDF page

```
import static com.foxit.sdk.pdf.PDFPage.*;  
...  
  
// Assuming PDFPage page has been loaded and parsed.  
  
// Flatten all contents of a PDF page.  
page.flatten(true, e_FlattenAll);  
  
// Flatten a PDF page without annotations.  
page.flatten(true, e_FlattenNoAnnot);  
  
// Flatten a PDF page without form controls.  
page.flatten(true, e_FlattenNoFormControl);  
  
// Flatten a PDF page without annotations and form controls (Equals to nothing to be flattened).  
page.flatten(true, e_FlattenNoAnnot | e_FlattenNoFormControl);  
...
```

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

```
import static com.foxit.sdk.pdf.PDFPage.*;  
...  
  
// Assuming PDFPage page has been loaded and parsed.  
  
...  
// Load the thumbnail bitmap. If the return value of function db.isEmpty() is true means no thumbnail can be found.  
Bitmap db = page.loadThumbnail();  
if (!db.isEmpty())  
{  
    int dbWidth = db.getWidth();  
    int dbHeight = db.getHeight();  
}  
  
// Set page thumbnail, db should be a valid bitmap.  
page.setThumbnail(db);  
...
```

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 [Renderer.setRenderContentFlags](#) to decide whether to render page and annotation both or not, and then use function [Renderer.startRender](#) to do the rendering. Function [Renderer.startQuickRender](#) can also be used to render page but only for thumbnail purpose.
- To render a single annotation, use function [Renderer.renderAnnot](#).
- To render on a bitmap, use function [Renderer.startRenderBitmap](#).
- To render a reflowed page, use function [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 [com.foxit.sdk.pdf.interform.Filler](#) object to fill the form, the function [pdf.interform.Filler.render](#) should be used to render the focused form control instead of the function [Renderer.renderAnnot](#).

Example:

3.4.1 How to render a page to a bitmap

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcrd.Matrix2D;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;

// Assuming PDFPage page has been loaded and parsed.

int width = (int) page.getWidth();
int height = (int) page.getHeight();
Matrix2D matrix = page.getDisplayMatrix(0, 0, width, height, page.getRotation());

// Prepare a bitmap for rendering.
Bitmap bitmap = new Bitmap(width, height, e_DIBArgb, null, 0);
bitmap.fillRect(0xFFFFFFFF, null);

// Render page.
Renderer render = new Renderer(bitmap, false);
render.startRender(page, matrix, null);
...
```

3.4.2 How to render page and annotation

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcr.Matrix2D;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;

// Assuming PDFPage page has been loaded and parsed.

int width = (int) page.getWidth();
int height = (int) page.getHeight();
Matrix2D matrix = page.getDisplayMatrix(0, 0, width, height, page.getRotation());

// Prepare a bitmap for rendering.
Bitmap bitmap = new Bitmap(width, height, e_DIBArgb, null, 0);
bitmap.fillRect(0xFFFFFFFF, null);

Renderer render(bitmap, false);
render.setRenderContentFlags(e_RenderAnnot | e_RenderPage);
render.startRender(page, matrix, null);
...
```

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

```
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.Attachments;
import com.foxit.sdk.pdf.FileSpec;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.objects.PDFNameTree;
...

PDFNameTree empty_nametree = new PDFNameTree();
{
    // Get information of attachments.
    Attachments attachments = new Attachments(doc, empty_nametree);
}
```

```
int count = attachments.getCount();
for (int i = 0; i < count; i++) {
    String key = attachments.getKey(i);

    FileSpec file_spec = attachments.getEmbeddedFile(key);
    if (!file_spec.isEmpty()) {
        String name = file_spec.getFileName();

        if (file_spec.isEmbedded()) {
            String export_file_path = output_path + name;
            file_spec.exportToFile(export_file_path);
        }
    }
}
```

3.5.2 How to remove all the attachments of a PDF

```
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.Attachments;

import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.objects.PDFNameTree;
...

// Assuming PDFDoc doc has been loaded.

PDFNameTree empty_nametree = new PDFNameTree();
{

    // Get information of attachments.
    Attachments attachments = new Attachments(doc, empty_nametree);
    attachments.removeAllEmbeddedFiles();
}
...
```

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 [TextPage](#) objects which are related to a specific page. [TextPage](#) class 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 [TextSearch](#) object with [TextPage](#) object.
- To access text such like hypertext link, construct a [PageTextLinks](#) object with [TextPage](#) object.

Example:

3.6.1 How to extract text from a PDF page

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.TextPage;
...
// Assuming PDFPage page has been loaded and parsed.

// Get the text page object.
TextPage textpage = new TextPage(page, e_ParseTextNormal);
int nCharCount = textpage.getCharCount();
String texts = textpage.getChars(0, nCharCount);
...
```

3.6.2 How to select text of a rectangle area in a PDF

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.TextPage;
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.common.fxcr.RectFArray;
...
// Assuming PDFPage page has been loaded and parsed.
...

TextPage textpage = new TextPage(page, e_ParseTextNormal);
RectF selRc = new RectF(100,100,250,250);
String selText = textpage.getTextInRect(selRc);
...
```

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 [TextSearch.setPattern](#), [TextSearch.setStartPage](#) (only useful for a text search in PDF document), [TextSearch.setEndPage](#) (only useful for a text search in PDF document) and [TextSearch.setSearchFlags](#).
- To do the searching, use function [TextSearch.findNext](#) or [TextSearch.findPrev](#).
- To get the searching result, use function [TextSearch.getMatchXXX\(\)](#).

Example:

3.7.1 How to search a text pattern in a PDF

```
import com.foxit.sdk.common.fxcr.RectF;
```

```
import com.foxit.sdk.common.fxcrf.RectFArray;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.TextSearch;
import com.foxit.sdk.pdf.TextPage;

...
TextSearch search = new TextSearch(doc, null, TextPage.e_ParseTextNormal );

int start_index = 0, end_index = doc.getPageCount() - 1;
search.setStartPage(0);
search.setEndPage(doc.getPageCount() - 1);

String pattern = "Foxit";
search.setPattern(pattern);

int flags = e_SearchNormal;
// if want to specify flags, you can do it like this:
// flags |= TextSearch::e_SearchMatchCase;
// flags |= TextSearch::e_SearchMatchWholeWord;
// flags |= TextSearch::e_SearchConsecutive;
search.setSearchFlags(flags);
int match_count = 0;
while (search.findNext()) {
    RectFArray rect_array = search.getMatchRects();
    match_count++;
}
...
```

3.8 Search and Replace

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

3.8.1 System requirements

Platform: Windows, Linux, Mac

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

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

SDK Version: Foxit PDF SDK 9.0 or higher

3.8.2 How to work with the search and replace function

```
import com.foxit.sdk.common.fxcrf.RectF;
import com.foxit.sdk.common.fxcrf.RectFArray;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.addon.pageeditor.*;

import static com.foxit.sdk.common.Constants.e_ErrSuccess;
```

```
PDFDoc doc = new PDFDoc(input_file);
error_code = doc.load(null);

// Instantiate a TextSearchReplace object.
TextSearchReplace searchreplace = new TextSearchReplace(doc);

// Configure search options, match whole words only, whether to set match only whole words and match case.
FindOption find_option = new FindOption();

ReplaceCallbackImpl replace_callback = new ReplaceCallbackImpl();

// Set replacing callback function.
searchreplace.setReplaceCallback(replace_callback);

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

// Replace with new text.
while(searchreplace.replaceNext("PDC")) {}
```

3.9 Text Link

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

Example:

3.9.1 How to retrieve hyperlinks in a PDF page

```
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.annots.*;
...

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

TextPage text_page = new TextPage(page, TextPage.e_ParseTextNormal);
PageTextLinks page_textlinks = new PageTextLinks(text_page);
TextLink text_link = page_textlinks.getTextLink(index); // specify an index.
String str_uri = text_link.getURI();
...
```

3.10 Bookmark

Foxit PDF SDK provides navigational tools called Bookmarks to allow users to quickly locate and link their point of interest within a PDF document. PDF bookmark is also called outline, and each

bookmark contains a destination or actions to describe where it links to. It is a tree-structured hierarchy, so function `pdf.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 `Bookmark.getFirstChild`.

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

- To access the parent bookmark, use function `Bookmark.getParent`.
- To access the first child bookmark, use function `Bookmark.getFirstChild`.
- To access the next sibling bookmark, use function `Bookmark.getNextSibling`.
- To insert a new bookmark, use function `Bookmark.insert`.
- To move a bookmark, use function `Bookmark.moveTo`.

Example:

3.10.1 How to find and list all bookmarks of a PDF

```
import com.foxit.sdk.pdf.Bookmark;
import com.foxit.sdk.pdf.PDFDoc;
...

// Assuming PDFDoc doc has been loaded.
...

Bookmark root = doc.getRootBookmark();
if (root.isEmpty()) {
    root = doc.createRootBookmark();
}

String titleStr = "";
Bookmark iterBookmark = root.getFirstChild ();
if (iterBookmark.isEmpty())
    return;
while (!iterBookmark.isEmpty())
{
    titleStr = iterBookmark.getTitle();
    if (iterBookmark.hasChild())
    {
        Bookmark childBookmark = iterBookmark.getFirstChild();
        titleStr = childBookmark.getTitle();
    }

    iterBookmark = iterBookmark.getNextSibling();
}
...
```

3.10.2 How to insert a new bookmark

```
import com.foxit.sdk.pdf.Bookmark;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.actions.Destination;
import static com.foxit.sdk.pdf.Bookmark.e_PosLastChild;

// Assuming PDFDoc doc has been loaded.

Bookmark root = doc.getRootBookmark();
if (root.isEmpty()) {
    root = doc.createRootBookmark();
}
Destination dest = Destination.createFitPage(doc, 0);

String ws_title = String.format("A bookmark to a page (index: %d)", 0);
Bookmark child = root.insert(ws_title, e_PosLastChild);
child.setDestination(dest);
child.setColor(0xF68C21);
```

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

```
import com.foxit.sdk.pdf.Bookmark;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.actions.Destination;
import static com.foxit.sdk.pdf.Bookmark.e_PosLastChild;

static void AddTOCToPDF(PDFDoc doc) throws PDFException {
    Int32Array intarray = new Int32Array();
    int depth = doc.getBookmarkLevelDepth();
    if (depth > 0) {
        for (int i = 1; i <= depth; i++) {
            intarray.add(i);
        }
    }
    String title = "";
    TableOfContentsConfig toc_config = new TableOfContentsConfig(title, intarray, true, false);

    // Add the table of contents
    doc.addTableOfContents(toc_config);
}
```

3.11 Form (AcroForm)

PDF currently supports two different forms for gathering information interactively from the user - AcroForms and XFA forms. Acroforms are the original PDF-based fillable forms, based on the PDF architecture. Foxit PDF SDK provides APIs to view and edit form field programmatically. Form fields are commonly used in PDF documents to gather data. The [Form](#) class 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 [Form.getFieldCount](#) and [Form.getField](#).
- To retrieve form controls from a PDF page, please use functions [Form.getControlCount](#) and [Form.getControl](#).
- To import form data from an XML file, please use function [Form.importFromXML](#); to export form data to an XML file, please use function [Form.exportToXML](#).
- To retrieve form filler object, please use function [Form.getFormFiller](#).

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

Example:

3.11.1 How to load the forms in a PDF

```
import com.foxit.sdk.pdf.interform.Form;
...

// Assuming PDFDoc doc has been loaded.
...

Boolean hasForm = doc.hasForm();
if(hasForm)
    Form form = new Form(doc);
...
```

3.11.2 How to count form fields and get the properties

```
import com.foxit.sdk.pdf.interform.Form;
import com.foxit.sdk.pdf.interform.Control;
import com.foxit.sdk.pdf.interform.Field;
...

// Assuming PDFDoc doc has been loaded.
...

Form form = new Form(doc);
String filter = "";

int nControlCount = form.getFieldCount(filter);
for (int i=0; i<nControlCount; i++)
{
    Field field = form.getField(i, filter);
    String fdName = field.getName();
    String fdValue = field.getValue();
    String DefaultValue = field.getDefaultValue();
}
...
```

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

```
import com.foxit.sdk.pdf.interform.Form;
...

// Assuming PDFDoc doc has been loaded.
...

Form form = new Form(doc);
form.exportToXML("form.xml");
...
```

3.11.4 How to import form data from a XML file

```
import com.foxit.sdk.pdf.interform.Form;
...

Form form = new Form(doc);
form.importFromXML("form.xml");
...
```

3.11.5 How to get coordinates of a form field

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

```
import com.foxit.sdk.common.Constants;
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.annots.Widget;
import com.foxit.sdk.pdf.interform.Control;
import com.foxit.sdk.pdf.interform.Field;
import com.foxit.sdk.pdf.interform.Form;
import com.foxit.sdk.pdf.PDFDoc;

...

// Load a document
PDFDoc doc = new PDFDoc(input_file);
int error_code = doc.load(null);
if (error_code != Constants.e_ErrSuccess) {
    System.out.println("The Doc " + input_file + " Error: " + error_code);
    return;
}

if (!doc.hasForm()) return;
Form form = new Form(doc);
for (int i = 0; i < form.getFieldCount(null); i++) {
    Field field = form.getField(i, null);
}
```

```

if (field.isEmpty()) continue;
for (int j = 0; j < field.getControlCount(); j++) {
    Control control = field.getControl(j);
    Widget widget = control.getWidget();
    //Get rectangle of the annot widget.
    RectF rect = widget.getRect();
}
}
...

```

3.12 XFA Form

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

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

Note:

- Foxit PDF SDK provides two callback classes [com.foxit.sdk.addon.xfa.AppProviderCallback](#) and [com.foxit.sdk.addon.xfa.DocProviderCallback](#) to represent the callback objects as an XFA document provider and an XFA application provider respectively. All the functions in those classes are used as callback functions. Pure virtual functions should be implemented by users.
- To use the XFA form feature, please make sure the license key has the permission of the 'XFA' module.

Example:

3.12.1 How to load XFADoc and represent an Interactive XFA form

```

import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.WStringArray;
import com.foxit.sdk.common.Constants;
import com.foxit.sdk.addon.xfa.*;
import com.foxit.sdk.addon.xfa.XFADoc.*;
import com.foxit.sdk.common.fxcr.*;
import com.foxit.sdk.common.WStringArray;
...
try {

```



```
// Create com.foxit.sdk.addon.xfa.AppProviderCallback handler.
XFAAppHandler xfa_app = new XFAAppHandler();
// Register it in application.
Library.registerXFAAppProviderCallback(xfa_app);

String input_file = input_path + "xfa_dynamic.pdf";
PDFDoc doc = new PDFDoc(input_file);
int error_code = doc.load(null);
if (error_code != e_ErrSuccess)
    return;

// Create com.foxit.sdk.addon.xfa.DocProviderCallback handler.
XFADocHandler xfa_dochandler = new XFADocHandler();
// Load xfa document from pdf document.
XFADoc xfa_doc = new XFADoc(doc, xfa_dochandler);
xfa_doc.startLoad(null);

} catch (PDFException e) {
    System.out.println(e.getMessage());
    return;
}
...
```

3.12.2 How to export and import XFA form data

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.WStringArray;
import com.foxit.sdk.common.Constants;
import com.foxit.sdk.addon.xfa.*;
import com.foxit.sdk.addon.xfa.XFADoc.*;
import com.foxit.sdk.common.fxcr.*;
import com.foxit.sdk.common.WStringArray;
...

// Assuming XFADoc xfa_doc has been loaded.

String output_xml = "xfa_form.xml";
xfa_doc.exportData(output_xml, XFADoc.e_ExportDataTypeXML);

xfa_doc.resetForm();
doc.saveAs("xfa_dynamic_resetform.pdf", PDFDoc.e_SaveFlagNormal);

xfa_doc.importData(output_xml);
doc.saveAs("xfa_dynamic_importdata.pdf", PDFDoc.e_SaveFlagNormal);
...
```

3.13 Form Design

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

Example:

3.13.1 How to add a text form field to a PDF

```
import com.foxit.sdk.pdf.interform.Form;
import com.foxit.sdk.pdf.interform.Control;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.common.fxcrf.RectF;
...

// Assuming PDFDoc doc has been loaded.
...

Form form = new Form(doc);
Control control = form.addControl(page, "Text Field0", Field.e_TypeTextField, new RectF(50, 600, 90, 640));
control.getField().setValue("3");

// Update text field's appearance.
control.getWidget().resetAppearanceStream();
...
```

3.13.2 How to remove a text form field from a PDF

```
import com.foxit.sdk.pdf.interform.Form;
import com.foxit.sdk.pdf.interform.Control;
import com.foxit.sdk.pdf.interform.Field;
...

// Assuming PDFDoc doc has been loaded.
...

Form form = new Form(doc);

int nControlCount = form.getFieldCount("Text Field0");
if (nControlCount > 0)
{
    Field field = form.getField(0, "Text Field0");
    form.removeField(field);
}
```

3.14 Annotations

3.14.1 General

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

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

Table 3-1

Annotation type	Description	Markup	Supported by SDK
Text(Note)	Text annotation	Yes	Yes
Link	Link Annotation	No	Yes
FreeText (TypeWriter/TextBox/Callout)	Free text annotation	Yes	Yes
Line	Line annotation	Yes	Yes
Square	Square annotation	Yes	Yes
Circle	Circle annotation	Yes	Yes
Polygon	Polygon annotation	Yes	Yes
PolyLine	PolyLine annotation	Yes	Yes
Highlight	Highlight annotation	Yes	Yes
Underline	Underline annotation	Yes	Yes
Squiggly	Squiggly annotation	Yes	Yes
StrikeOut	StrikeOut annotation	Yes	Yes
Stamp	Stamp annotation	Yes	Yes
Caret	Caret annotation	Yes	Yes
Ink(pencil)	Ink annotation	Yes	Yes
Popup	Popup annotation	No	Yes
File Attachment	FileAttachment annotation	Yes	Yes
Sound	Sound annotation	Yes	No
Movie	Movie annotation	No	No

Widget*	Widget annotation	No	Yes
Screen	Screen annotation	No	Yes
PrinterMark	PrinterMark annotation	No	No
TrapNet	Trap network annotation	No	No
Watermark*	Watermark annotation	No	Yes
3D	3D annotation	No	No
Redact	Redact annotation	Yes	Yes

Note:

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

Example:

3.14.1.1 How to add a link annotation to a PDF page

```
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.annots.Link;
...

// Assuming PDFPage page has been loaded and parsed.

// Add link annotation
Link link = new Link(page.addAnnot(Annot.e_Link, new RectF(350, 350, 380, 400)));
link.setHighlightingMode(Annot.e_HighlightingToggle);

// Appearance should be reset.
link.resetAppearanceStream();
...
```

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

```
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.annots.Highlight;
import com.foxit.sdk.common.fxcrp.PointF;
import com.foxit.sdk.common.fxcrp.PointFArray;
import com.foxit.sdk.common.fxcrp.RectF;
import com.foxit.sdk.pdf.annots.QuadPoints;
import com.foxit.sdk.pdf.annots.QuadPointsArray;
...
```

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

// Add highlight annotation
Highlight highlight = new Highlight(page.addAnnot(Annot.e_Highlight, new RectF(10, 450, 100, 550)));
highlight.setContent("Highlight");
QuadPoints quad_points = new QuadPoints();
quad_points.setFirst(new PointF(10, 500));
quad_points.setSecond(new PointF(90, 500));
quad_points.setThird(new PointF(10, 480));
quad_points.setFourth(new PointF(90, 480));
QuadPointsArray quad_points_array = new QuadPointsArray();
quad_points_array.add(quad_points);
highlight.setQuadPoints(quad_points_array);
highlight.setSubject("Highlight");
highlight.setTitle("Foxit SDK");
highlight.setCreationDateTime(GetLocalDateTime());
highlight.setModifiedDateTime(GetLocalDateTime());
highlight.setUniquelD(RandomUID());

// Appearance should be reset.
highlight.resetAppearanceStream();

...
```

3.14.1.3 How to set the popup information when creating markup annotations

```
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.annots.Note;
import com.foxit.sdk.pdf.annots.Popup;
import com.foxit.sdk.common.fxcr.RectF;
...

// Assuming PDFPage page has been loaded and parsed.

// Add note annotation
Note note = new Note(page.addAnnot(Annot.e_Note, new RectF(10, 350, 50, 400)));
note.setIconName("Comment");
note.setSubject("Note");
note.setTitle("Foxit SDK");
note.setContent("Note annotation.");
note.setCreationDateTime(GetLocalDateTime());
note.setModifiedDateTime(GetLocalDateTime());
note.setUniquelD(RandomUID());

// Add popup to note annotation
Popup popup = new Popup(page.addAnnot(Annot.e_Popup, new RectF(300, 450, 500, 550)));
popup.setBorderColor(0x00FF00);
popup.setOpenStatus(false);
popup.setModifiedDateTime(GetLocalDateTime());
note.setPopup(popup);

// Appearance should be reset.
```

```
note.resetAppearanceStream();  
...
```

3.14.1.4 How to get page transformation matrix between PDF page coordinates and rendering device coordinates

```
import com.foxit.sdk.pdf.PDFPage;  
import com.foxit.sdk.common.fxcr.RectF;  
import com.foxit.sdk.common.fxcr.Matrix2D;  
  
// Assuming PDFDoc doc has been loaded.  
// Assuming PDFPage page has been loaded and parsed.  
...  
  
Matrix2D mtx = page.getDisplayMatrix(0, 0, width, height, page.getRotation());  
  
RectF srcRc = new RectF(100, 200, 200, 100);  
RectF pageRc = new RectF(100, 200, 200, 100);  
mtx.transformRect(pageRc);  
  
RectF devRc = new RectF(pageRc.getLeft(), pageRc.getBottom(), pageRc.getRight(), pageRc.getTop());  
Matrix2D reverse_mtx = new Matrix2D();  
reverse_mtx.setReverse(mtx);  
  
reverse_mtx.transformRect(devRc);  
...
```

3.14.1.5 How to extract the texts under text markup annotations

```
import com.foxit.sdk.PDFException;  
import com.foxit.sdk.common.Library;  
import com.foxit.sdk.pdf.PDFDoc;  
import com.foxit.sdk.pdf.PDFPage;  
import com.foxit.sdk.pdf.TextPage;  
import com.foxit.sdk.pdf.annotations.Annot;  
import com.foxit.sdk.pdf.annotations.TextMarkup;  
  
import static com.foxit.sdk.common.Constants.e_ErrSuccess;  
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;  
  
// Assuming PDFDoc doc has been loaded.  
  
PDFPage page = doc.getPage(0);  
// Parse the first page.  
page.startParse(e_ParsePageNormal, null, false);  
// Get a TextPage object.  
TextPage text_page = new TextPage(page, TextPage.e_ParseTextNormal);  
  
int annot_count = page.getAnnotCount();  
for (int i = 0; i < annot_count; i++) {  
    Annot annot = page.getAnnot(i);  
    TextMarkup text_markup = new TextMarkup(annot);  
    if (!text_markup.isEmpty()) {
```

```
// Get the texts which intersect with a text markup annotation.
String text = text_page.getTextUnderAnnot(text_markup);
}
}
```

3.14.1.6 How to add richtext for freetext annotation

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.Font;
import com.foxit.sdk.pdf.annots.FreeText;
import com.foxit.sdk.pdf.annots.RichTextStyle;

// 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.
Annot annot = pdf_page.addAnnot(Annot.e_FreeText, new RectF(50, 50, 150, 100));
FreeText freetext = new FreeText(annot);
// Set annotation's properties.

// Add/insert richtext string with style.
RichTextStyle richtext_style = new RichTextStyle();
richtext_style.setFont(new Font("Times New Roman", 0, Font.e_CharsetANSI, 0));
richtext_style.setText_color(0xFF0000);
richtext_style.setText_size(10);
freetext.addRichText("Textbox annotation ", richtext_style);

richtext_style.setText_color(0x00FF00);
richtext_style.setIs_underline(true);
freetext.addRichText("1-underline ", richtext_style);

richtext_style.setFont(new Font("Calibri", 0, Font.e_CharsetANSI, 0));
richtext_style.setText_color(0x0000FF);
richtext_style.setIs_underline(false);
richtext_style.setIs_strikethrough(true);
int richtext_count = freetext.getRichTextCount();
freetext.insertRichText(richtext_count-1, "2_strikethrough ", richtext_style);

// Appearance should be reset.
freetext.resetAppearanceStream();
```

3.14.2 Import annotations from or export annotations to a FDF file

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

Example:

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

```
import com.foxit.sdk.common.Range;
```

```
import com.foxit.sdk.fdf.FDFDoc;
import com.foxit.sdk.pdf.PDFDoc;
import static com.foxit.sdk.pdf.PDFDoc.e_Annots;

// Assuming PDFDoc doc has been loaded.
...

Range empty_range = new Range();
{
    String input_file = input_path + "AboutFoxit.pdf";
    String fdf_file = input_path + "AnnotationData.fdf";
    PDFDoc pdf_doc = new PDFDoc(input_file);
    error_code = pdf_doc.load(null);
    if (error_code != e_ErrSuccess) {
        System.out.println("The Doc " + input_file + " Error: " + error_code);
        return;
    }
    FDFDoc fdf_doc = new FDFDoc(fdf_file);
    pdf_doc.importFromFDF(fdf_doc, e_Annots, empty_range);
}
...
```

3.15 Image Conversion

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

Example:

3.15.1 How to convert PDF pages to bitmap files

```
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcrd.Matrix2D;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;

// Assuming PDFDoc doc has been loaded.
...

Image image = new Image();

// Get page count
```



```
int nPageCount = doc.getPageCount();
for (int i = 0; i < nPageCount; i++) {
    PDFPage page = doc.getPage(i);
    // Parse page.
    page.startParse(e_ParsePageNormal, null, false);

    int width = (int) page.getWidth();
    int height = (int) page.getHeight();
    Matrix2D matrix = page.getDisplayMatrix(0, 0, width, height, page.getRotation());

    // Prepare a bitmap for rendering.
    Bitmap bitmap = new Bitmap(width, height, e_DIBArgb, null, 0);
    bitmap.fillRect(0xFFFFFFFF, null);

    // Render page.
    Renderer render = new Renderer(bitmap, false);
    render.startRender(page, matrix, null);
    image.addFrame(bitmap);
}
...
```

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

```
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcrd.Matrix2D;
import com.foxit.sdk.pdf.PDFPage;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;

...
// Parse page.
page.startParse(e_ParsePageNormal, null, false);
int width = (int) page.getWidth();
int height = (int) page.getHeight();
int render_sum = 10;
int width_scale = 1;
int height_scale = 1;
int little_width = width * width_scale;
int little_height = height / render_sum * height_scale;
for(int i = 0; i < render_sum; i++) {
    // According to Matrix, do module rendering for large PDF files.
    Matrix2D matrix = page.GetDisplayMatrix(0, -1 * i * little_height, little_width, height * height_scale,
    page.GetRotation());
    // Prepare a bitmap for rendering.
    Bitmap bitmap = new Bitmap(little_width, little_height, e_DIBArgb, null, 0);
    bitmap.fillRect(0xFFFFFFFF, null);
    // Render page.
}
```

```
Renderer render = new Renderer(bitmap, false);
render.startRender(page, matrix, null);
// The bitmap data will be added to the end of image file after rendering.
...
}
```

3.15.2 How to convert a image file to PDF file

```
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.fxcrf.PointF;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;

import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;
import static com.foxit.sdk.pdf.PDFPage.e_SaveFlagNoOriginal;

Image image = new Image(input_file);
int count = image.getFrameCount();

PDFDoc doc = new PDFDoc();
for (int i = 0; i < count; i++) {
    PDFPage page = doc.insertPage(i, PDFPage.e_SizeLetter);
    page.startParse(e_ParsePageNormal, null, false);
    // Add image to page.
    page.addImage(image, i, new PointF(0, 0), page.getWidth(), page.getHeight(), true);
}

doc.saveAs("convertedPDF.pdf", e_SaveFlagNoOriginal);
...
```

3.16 Watermark

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

Example:

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

```
import com.foxit.sdk.common.Font;
import com.foxit.sdk.pdf.*;
import com.foxit.sdk.pdf.Watermark;
import com.foxit.sdk.pdf.WatermarkSettings;
import com.foxit.sdk.pdf.WatermarkTextProperties;

import com.foxit.sdk.common.Constants;

import static com.foxit.sdk.common.Constants.e_AlignmentCenter;
```

```
import static com.foxit.sdk.common.Font.e_StdIDTimesB;  
import static com.foxit.sdk.pdf.WatermarkSettings.*;  
import static com.foxit.sdk.pdf.WatermarkTextProperties.e_FontStyleNormal;  
...  
  
// Assuming PDFDoc doc has been loaded.  
  
WatermarkSettings settings = new WatermarkSettings();  
settings.setFlags(e_FlagASPageContents | e_FlagOnTop);  
settings.setOffset_x(0);  
settings.setOffset_y(0);  
settings.setOpacity(90);  
settings.setPosition(Constants.e_PosTopRight);  
settings.setRotation(-45.f);  
settings.setScale_x(1.f);  
settings.setScale_y(1.f);  
  
WatermarkTextProperties text_properties = new WatermarkTextProperties();  
text_properties.setAlignment(e_AlignmentCenter);  
text_properties.setColor(0xF68C21);  
text_properties.setFont_style(e_FontStyleNormal);  
text_properties.setLine_space(1);  
text_properties.setFont_size(12.f);  
text_properties.setFont(new Font(e_StdIDTimesB));  
  
Watermark watermark = new Watermark(doc, "Foxit PDF SDK\nwww.foxitsoftware.com", text_properties,  
settings);  
watermark.insertToPage(page);  
  
// Save document to file  
...
```

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

```
import com.foxit.sdk.common.Bitmap;  
import com.foxit.sdk.common.Image;  
import com.foxit.sdk.pdf.*;  
import com.foxit.sdk.pdf.Watermark;  
import com.foxit.sdk.pdf.WatermarkSettings;  
  
import com.foxit.sdk.common.Constants;  
  
import static com.foxit.sdk.common.Constants.e_AlignmentCenter;  
import static com.foxit.sdk.common.Font.e_StdIDTimesB;  
import static com.foxit.sdk.pdf.WatermarkSettings.*;  
import static com.foxit.sdk.pdf.WatermarkTextProperties.e_FontStyleNormal;  
...  
  
// Assuming PDFDoc doc has been loaded.  
  
WatermarkSettings settings = new WatermarkSettings();  
settings.setFlags(e_FlagASPageContents | e_FlagOnTop);
```

```
settings.setOffset_x(0.f);
settings.setOffset_y(0.f);
settings.setOpacity(20);
settings.setPosition(Constants.e_PosCenter);
settings.setRotation(0.0f);

Image image = new Image(image_file);
Bitmap bitmap = image.getFrameBitmap(0);
settings.setScale_x(page.getWidth() * 0.618f / bitmap.getWidth());
settings.setScale_y(settings.getScale_x());

Watermark watermark = new Watermark(doc, image, 0, settings);
watermark.insertToPage(page);

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

3.16.3 How to remove all watermarks from a page

```
import com.foxit.sdk.pdf.PDFPage;
...
// Assuming PDFPage page has been loaded and parsed.
...
page.removeAllWatermarks();
...
// Save document to file
...
```

3.17 Barcode

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

Table 3-2

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

Example:

3.17.1 How to generate a barcode bitmap from a string

```
import com.foxit.sdk.common.Barcode;
...

// Strings used as barcode content.
String codeStr = "TEST-SHEET"

// Barcode format types.
int codeFormat = Barcode.e_FormatCode39;

// Format error correction level of QR code.
int qrLevel = Barcode.e_QRCorrectionLevelLow;

// Image names for the saved image files for QR code.
String bmpQrName = "/QR_CODE_TestForBarcodeQrCode_L.bmp";

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

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

Barcode barcode = new Barcode();
Bitmap bitmap = barcode.generateBitmap(codeStr, codeFormat, unitWidth, unitHeight, qrLevel);
...
```

3.18 Security

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

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

Example:

3.18.1 How to encrypt a PDF file with Certificate

```
import com.foxit.sdk.pdf.CertificateEncryptData;
import com.foxit.sdk.pdf.CertificateSecurityCallback;
import com.foxit.sdk.pdf.CertificateSecurityHandler;
import java.io.File;
import java.io.UnsupportedEncodingException;
```

```
import java.security.MessageDigest;
import java.util.ArrayList;
import java.util.Random;
import java.io.ByteArrayOutputStream;
import java.io.FileInputStream;
import java.util.Enumeration;
import java.security.Key;
import java.security.KeyStore;
import java.security.cert.CertificateFactory;
import javax.crypto.Cipher;
...
// Assuming PDFDoc doc has been loaded.
...

public static Key getPublicKey(String cerPath) {
    try {
        CertificateFactory certificateFactory = CertificateFactory.getInstance("X.509");
        FileInputStream stream = new FileInputStream(cerPath);
        java.security.cert.Certificate certificate = certificateFactory.generateCertificate(stream);
        stream.close();
        return certificate.getPublicKey();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}

private static byte[] cryptByKey(byte[] inputData, Key key, int opmode) {
    if (inputData == null) return null;
    // The max length of decrypted byte array: 128
    final int max_crypt_block = 128;
    try {
        Cipher cipher = Cipher.getInstance(key.getAlgorithm());
        cipher.init(opmode, key);
        int len = inputData.length;
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        int offSet = 0;
        byte[] data;
        // Decrypt data segment by segment
        while (len > offSet) {
            data = cipher.doFinal(inputData, offSet,
                (len - offSet > max_crypt_block) ? max_crypt_block : (len - offSet));
            stream.write(data, 0, data.length);
            offSet += max_crypt_block;
        }
        byte[] outputData = stream.toByteArray();
        stream.close();
        return outputData;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}
```

```
}

public static byte[] encryptByKey(byte[] plainData, Key key) {
    return cryptByKey(plainData, key, Cipher.ENCRYPT_MODE);
}

public class CertificateSecurityEvent extends CertificateSecurityCallback {
    private String filePath;
    private String password;

    public CertificateSecurityEvent(String filePath, String password) {
        this.filePath = filePath;
        this.password = password;
    }

    @Override
    public void release() {}

    @Override
    public byte[] getDecryptionKey(byte[] arg0) {
        return CryptUtil.decryptByKey(arg0, CryptUtil.getPrivateKey(filePath, password));
    }
}

Random rand = new Random(23);
byte[] seed = new byte[24];
rand.nextBytes(seed);
for (int i = 20; i < 24; i++) {
    seed[i] = (byte) 0xFF;
}

PDFDoc doc = new PDFDoc(input_file);
int error_code = doc.load(null);
if (error_code != e_ErrSuccess) {
    System.out.println("The Doc " + input_file + " Error: " + error_code);
    return;
}

// Do encryption.
String cert_file_path = input_path + "foxit.cer";
ArrayList<byte[]> envelopes = new ArrayList<byte[]>();
byte[] bytes=null;
try {
    bytes = CryptUtil.encryptByKey(seed, CryptUtil.getPublicKey(cert_file_path));
    envelopes.add(bytes);
} catch (Exception e) {
    System.out.println("[Failed] Cannot get certificate information from " + cert_file_path);
    return;
}
byte[] data=new byte[20+bytes.length];
System.arraycopy(seed, 0, data, 0, 20);
System.arraycopy(bytes, 0, data, 20, bytes.length);
```

```
MessageDigest messageDigest = MessageDigest.getInstance("SHA1");
messageDigest.update(data);
byte[] initial_key = new byte[16];
System.arraycopy(messageDigest.digest(),0,initial_key,0,16);
CertificateSecurityHandler handler = new CertificateSecurityHandler();
CertificateEncryptData encrypt_data = new CertificateEncryptData(true, SecurityHandler.e_CipherAES,
envelopes);
handler.initialize(encrypt_data, initial_key);

doc.setSecurityHandler(handler);
String output_file = output_directory + "certificate_encrypt.pdf";
doc.saveAs(output_file, PDFDoc.e_SaveFlagNoOriginal);
...
```

3.18.2 How to encrypt a PDF file with Foxit DRM

```
import com.foxit.sdk.pdf.DRMSecurityCallback;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.SecurityHandler;
...

public class DRMSecurityEvent extends DRMSecurityCallback {
    private String fileID;
    private byte[] initialKey;

    public DRMSecurityEvent(String fileID, byte[] initialKey) {
        this.fileID = fileID;
        this.initialKey = initialKey;
    }

    @Override
    public void release() {}

    @Override
    public int getCipherType(PDFDoc arg0, String arg1) {
        return SecurityHandler.e_CipherAES;
    }

    @Override
    public String getFileID(PDFDoc arg0, String arg1) {
        return fileID;
    }

    @Override
    public byte[] getInitialKey(PDFDoc arg0, String arg1) {
        return initialKey;
    }

    @Override
    public int getKeyLength(PDFDoc arg0, String arg1) {
        return 16;
    }
}
```



```
@Override
public int getUserPermissions(PDFDoc arg0, String arg1) {
    return 0xFFFFFFFF;
}

@Override
public boolean isOwner(PDFDoc arg0, String arg1) {
    return true;
}

}

PDFDoc doc = new PDFDoc(input_file);
int error_code = doc.load(null);
if (error_code != e_ErrSuccess) {
    return;
}

// Do encryption.
DRMSecurityHandler handler = new DRMSecurityHandler();
String file_id = "Simple-DRM-file-ID";
String initialize_key = "Simple-DRM-initialize-key";
DRMEncryptData encrypt_data = new DRMEncryptData(true, "Simple-DRM-filter", SecurityHandler.e_CipherAES,
16, true, 0xffffffff);
handler.initialize(encrypt_data, file_id, initialize_key);
doc.setSecurityHandler(handler);

String output_file = output_directory + "foxit_drm_encrypt.pdf";
doc.saveAs(output_file, PDFDoc.e_SaveFlagNoOriginal);
...
```

3.19 Reflow

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

Example:

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

```
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Renderer;
import com.foxit.sdk.common.fxcrd.Matrix2D;
import com.foxit.sdk.common.fxcrd.PointF;
import com.foxit.sdk.common.fxcrd.RectF;
import com.foxit.sdk.common.fxcrd.RectI;
```

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.ReflowPage;

import static com.foxit.sdk.common.Bitmap.e_DIBArgb;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import static com.foxit.sdk.pdf.PDFPage.e_ParsePageNormal;
import static com.foxit.sdk.pdf.ReflowPage.e_Normal;
import static com.foxit.sdk.pdf.ReflowPage.e_WithImage;
...

public static void SaveBitmap(Bitmap bitmap, int index, String file_name) throws PDFException {
    RectF margin = new RectF(50, 30, 30, 30);
    PointF size = new PointF(480, 800);
    RectI rect = new RectI(0, 0, (int) size.getX(), (int) margin.getTop());
    bitmap.fillRect(0xFFFFFFFF, rect);
    rect = new RectI(0, (int) size.getY() - (int) margin.getBottom(), (int) size.getX(), (int) size.getY());
    bitmap.fillRect(0xFFFFFFFF, rect);
    Image image = new Image();
    image.addFrame(bitmap);

    String save_path;
    String sIndex;
    sIndex = "" + index;
    save_path = output_path + "reflow" + file_name + sIndex + ".bmp";
    image.saveAs(save_path);
}

RectF margin = new RectF(50, 30, 30, 30);
PointF size = new PointF(480, 800);
int nCount = doc.getPageCount();
for (int i = 0; i < nCount; i++) {
    PDFPage page = doc.getPage(i);
    // Parse PDF page.
    page.startParse(e_ParsePageNormal, null, false);

    ReflowPage reflow_page = new ReflowPage(page);
    // Set some arguments used for parsing the reflow page.
    reflow_page.setLineSpace(0);
    reflow_page.setScreenMargin((int) margin.getLeft(), (int) margin.getTop(), (int) margin.getRight(), (int)
margin.getBottom());
    reflow_page.setScreenSize(size.getX(), size.getY());
    reflow_page.setZoom(100);
    reflow_page.setParseFlags(e_Normal);

    // Parse reflow page.
    reflow_page.startParse(null);

    // Get actual size of content of reflow page. The content size does not contain the margin.
    float content_width = reflow_page.getContentWidth();
    float content_height = reflow_page.getContentHeight();

    // Create a bitmap for rendering the reflow page. The bitmap size contains the margin.
```

```
Bitmap bitmap = new Bitmap((int) (content_width + margin.getLeft() + margin.getRight()),
    (int) (content_height + margin.getTop() + margin.getBottom()), e_DIBArgb, null, 0);
bitmap.fillRect(0xFFFFFFFF, null);

// Render reflow page.
Renderer renderer = new Renderer(bitmap, false);
Matrix2D matrix = reflow_page.getDisplayMatrix(0, 0);
renderer.startRenderReflowPage(reflow_page, matrix, null);
String file_name = "_single_";
SaveBitmap(bitmap, i, file_name);
}
...
```

3.20 Asynchronous PDF

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

3.21 Pressure Sensitive Ink

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

Example:

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

```
import com.foxit.sdk.pdf.PSI;
import com.foxit.sdk.common.fxcrd.PointF;
...

PSI psi = new PSI(480, 180, true);

// Set ink diameter.
psi.setDiameter(9);
```

```
// Set ink color.
psi.setColor(0x434236);

// Set ink opacity.
psi.setOpacity(0.8f);

// Add points to pressure sensitive ink.
float x = 121.3043f;
float y = 326.6846f;
float pressure = 0.0966f;
int type = 1;
PointF point = new PointF(x, y);
psi.addPoint(point, type, pressure);
...
```

3.22 Wrapper

Wrapper provides a way for users to save their own data related to a PDF document. For example, when opening an encrypted unauthorized PDF document, users may get an error message. In this case, users can still access wrapper data even when they do not have permissions to the PDF content. The wrapper data could be used to provide information like where to get decryption method of this document.

Example:

3.22.1 How to open a document including wrapper data

```
import com.foxit.sdk.common.fxcrf.FileReaderCallback;

import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.WrapperData;
...

PDFDoc doc = new PDFDoc(file_name);
int code = doc.load(null);
if (code != e_ErrSuccess) {
    return;
}
if (!doc.isWrapper()) {
    return;
}
int offset = doc.getWrapperOffset();
FileReader file_reader = new FileReader(offset);
file_reader.LoadFile(file_name);

PDFDoc doc_real = new PDFDoc(file_reader, false);
code = doc_real.load(null);
if (code != e_ErrSuccess) {
    return;
}
```

```
}  
...
```

3.23 PDF Objects

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

Example:

3.23.1 How to remove some properties from catalog dictionary

```
import com.foxit.sdk.pdf.PDFDoc;  
import com.foxit.sdk.pdf.objects.PDFDictionary;  
...  
  
PDFDictionary catalog = document.getCatalog();  
if (null == catalog) {  
    return;  
}  
  
String[] key_strings = {"Type", "Boolean", "Name", "String", "Array", "Dict"};  
int count = key_strings.length;  
for (int i = 0; i < count; i++) {  
    if (catalog.containsKey(key_strings[i])) {  
        catalog.removeAt(key_strings[i]);  
    }  
}  
...  
...
```

3.24 Page Object

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

Example:

3.24.1 How to create a text object in a PDF page

```
import com.foxit.sdk.common.fxcrd.Matrix2D;
```

```
import com.foxit.sdk.common.fxcrf.PointF;
import com.foxit.sdk.common.fxcrf.RectF;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.graphics.*;
import com.foxit.sdk.common.Font;
import static com.foxit.sdk.common.Font.*;
...

long position = page.getLastGraphicsObjectPosition(e_TypeText);
TextObject text_object = TextObject.create();

text_object.setFill(0xFFFF7F00);

// Prepare text state
TextState state = new TextState();
state.setFont_size(80.0f);
state.setFont(new Font("Simsun", e_StylesSmallCap, e_CharsetGB2312, 0));
state.setTextmode(e_ModeFill);
text_object.setTextState(page, state, false, 750);

// Set text.
text_object.setText("Foxit Software");
long last_position = page.insertGraphicsObject(position, text_object);

RectF rect = text_object.getRect();
float offset_x = (page.getWidth() - (rect.getRight() - rect.getLeft())) / 2;
float offset_y = page.getHeight() * 0.8f - (rect.getTop() - rect.getBottom()) / 2;
text_object.transform(new Matrix2D(1, 0, 0, 1, offset_x, offset_y), false);

// Generate content
page.generateContent();
...
```

3.24.2 How to add an image logo to a PDF page

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.graphics.*;
import com.foxit.sdk.common.Font;
import static com.foxit.sdk.common.Font.*;
import com.foxit.sdk.common.Image;
...

long position = page.getLastGraphicsObjectPosition(e_TypeImage);
Image image = new Image(image_file);
ImageObject image_object = ImageObject.create(page.getDocument());
image_object.setImage(image, 0);

float width = image.getWidth();
float height = image.getHeight();
```

```
float page_width = page.getWidth();
float page_height = page.getHeight();

// Please notice the matrix value.
image_object.setMatrix(new Matrix2D(width, 0, 0, height, (page_width - width) / 2.0f, (page_height - height) / 2.0f));

page.insertGraphicsObject(position, image_object);
page.generateContent();
...
```

3.25 Marked content

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

Example:

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

```
import com.foxit.sdk.pdf.graphics.GraphicsObject;
import com.foxit.sdk.pdf.graphics.MarkedContent;
...

long position = page.getFirstGraphicsObjectPosition(e_TypeText);
GraphicsObject text_obj = page.getGraphicsObject(position);
MarkedContent content = text_obj.getMarkedContent();

int nCount = content.getItemCount();
// Get marked content property
for (int i = 0; i < nCount; i++) {
    String tag_name = content.getItemTagName(i);
    int mcid = content.getItemMCID(i);
}
...
```

3.26 Layer

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

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

Example:

3.26.1 How to create a PDF layer

```
import static com.foxit.sdk.pdf.LayerTree;
import static com.foxit.sdk.pdf.LayerNode;
...

LayerTree layertree = new LayerTree(doc);
LayerNode root = layertree.getRootNode();
```

3.26.2 How to set all the layer nodes information

```
import com.foxit.sdk.pdf.*;
...

static void setAllLayerNodesInformation(LayerNode layer_node) throws PDFException {
    if (layer_node.hasLayer()) {
        layer_node.setDefaultVisible(true);
        layer_node.setExportUsage(e_StateUndefined);
        layer_node.setViewUsage(e_StateOFF);
        LayerPrintData print_data = new LayerPrintData("subtype_print", e_StateON);
        layer_node.setPrintUsage(print_data);
        LayerZoomData zoom_data = new LayerZoomData(1, 10);
        layer_node.setZoomUsage(zoom_data);
        String new_name = String.format("[View_OFF_Print_ON_Export_Undefined]") + layer_node.getName();
        layer_node.setName(new_name);
    }
    int count = layer_node.getChildrenCount();
    for (int i = 0; i < count; i++) {
        LayerNode child = layer_node.getChild(i);
        setAllLayerNodesInformation(child);
    }
}
...
```

3.26.3 How to edit layer tree

```
import com.foxit.sdk.pdf.*;
...

LayerTree layertree = new LayerTree(doc);
LayerNode root = layertree.getRootNode();
```



```
int children_count = root.getChildrenCount();
root.removeChild(children_count - 1);
LayerNode child = root.getChild(children_count - 2);
LayerNode child0 = root.getChild(0);
child.moveTo(child0, 0);
child.addChild(0, "AddedLayerNode", true);
child.addChild(0, "AddedNode", false);
...
```

3.27 Signature

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

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

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

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

Example:

3.27.1 How to sign the PDF document with a signature

```
import com.foxit.sdk.pdf.*;
...

String filter = "Adobe.PPKLite";
String sub_filter = "adbe.pkcs7.detached";
PDFPage pdf_page = pdf_doc.getPage(0);
// Add a new signature to first page.
com.foxit.sdk.pdf.Signature new_signature = AddSignature(pdf_page, sub_filter);
// Set filter and subfilter for the new signature.
new_signature.setFilter(filter);
new_signature.setSubFilter(sub_filter);
boolean is_signed = new_signature.isSigned();
int sig_state = new_signature.getState();
String signed_pdf_path = output_directory + "signed_newssignature.pdf";

String cert_file_path = input_path + "foxit_all.pfx";
```

```
String cert_file_password = "123456";

// Cert file path will be passed back to application through callback function SignatureCallback::Sign().
// In this demo, the cert file path will be used for signing in callback function SignatureCallback::Sign().
new_signature.startSign(cert_file_path, cert_file_password.getBytes(), e_DigestSHA1, signed_pdf_path, null,
null);
...
```

3.27.2 How to implement signature callback function of signing

```
import com.foxit.sdk.pdf.*;
...

// Implementation of SignatureCallback
class SignatureCallbackImpl extends SignatureCallback {
    private String sub_filter_;
    private DigestContext digest_context_ = null;
    byte[] arrall = null;

    SignatureCallbackImpl(String subfilter) {
        sub_filter_ = subfilter;
    }

    @Override
    public boolean startCalcDigest(FileReaderCallback var1, int[] var2,
        com.foxit.sdk.pdf.Signature var3, Object var4) {
        digest_context_ = new DigestContext(var1, var2, var2.length);
        return true;
    }

    @Override
    public int continueCalcDigest(Object var1, PauseCallback var2) {
        return com.foxit.sdk.common.Progressive.e_Finished;
    }

    @Override
    public byte[] getDigest(Object var1) {
        return arrall;
    }

    @Override
    public byte[] sign(byte[] digest, String cert_path, byte[] cert_password, int digest_algorithm,
        java.lang.Object client_data){
        String encryptStr = null;

        try {
            try {
                FileReaderCallback filehandler = digest_context_.file_read_callback_;
                {
                    long size = filehandler.getSize();
                    byte[] arr1 = new byte[digest_context_.byte_range_array[1]];
                    filehandler.readBlock(arr1,
```

```

        digest_context_.byte_range_array_[0],
        digest_context_.byte_range_array_[1]);
        byte[] arr2 = new byte[digest_context_.byte_range_array_[3]];
        filehandler.readBlock(arr2,
            digest_context_.byte_range_array_[2],
            digest_context_.byte_range_array_[3]);

        size = 0;
        arrall = new byte[(int) digest_context_.byte_range_array_[1]
            + (int) digest_context_.byte_range_array_[3]];
        System.arraycopy(arr1, 0, arrall, 0, arr1.length);
        System.arraycopy(arr2, 0, arrall, arr1.length, arr2.length);
    }

    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    encryptStr = CertUtil.SignMsg(arrall, signature.input_path
        + "foxit_all.pfx", "123456");
    return encryptStr.getBytes();

    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return null;
}

@Override
public byte[] sign(byte[] digest, StreamCallback cert_path, byte[] cert_password, int digest_algorithm,
java.lang.Object client_data){
    return null;
}

@Override
public int verifySigState(byte[] var1, byte[] var2, Object var3) {
    byte[] arrall_verify = null;
    boolean verify_state = false;
    try {
        verify_state = CertUtil.VerifyMsg(new String(var2).toLowerCase(), arrall,
            signature.input_path + "foxit.cer");
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return verify_state ? com.foxit.sdk.pdf.Signature.e_StateVerifyNoChange :
com.foxit.sdk.pdf.Signature.e_StateVerifyChange;
}

@Override
public boolean isNeedPadData() {return false;}

```

```
}  
...
```

3.28 Long term validation (LTV)

From version 7.0, Foxit PDF SDK provides APIs to establish long term validation (LTV) of signatures, which is mainly used to solve the verification problem of signatures that have already expired. LTV requires DSS (Document Security Store) which contains the verification information of the signatures, as well as DTS (Document Timestamp Signature) which belongs to the type of time stamp signature.

In order to support LTV, Foxit PDF SDK provides:

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

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

Example:

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

```
// 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.  
TimeStampServerMgr.initialize();  
TimeStampServer timestamp_server = TimeStampServerMgr.addServer(server_name, server_url,  
server_username, server_password);  
TimeStampServerMgr.setDefaultServer(timestamp_server);  
  
// Assume that "signed_pdf_path" represents a signed PDF document which contains signed signature.  
PDFDoc pdf_doc = new PDFDoc(signed_pdf_path);  
pdf_doc.startLoad(null, false, null);  
{
```

```
// Use LTVVerifier to verify and add DSS.
LTVVerifier ltv_verifier = new LTVVerifier(pdf_doc, true, true, false, LTVVerifier.e_SignatureCreationTime);
// Set verifying mode which is necessary.
ltv_verifier.setVerifyMode(LTVVerifier.e_VerifyModeAcrobat);
SignatureVerifyResultArray sig_verify_result_array = ltv_verifier.verify();
for (long i = 0; i < sig_verify_result_array.getSize(); i++) {
    // ltv state would be e_LTVStateNotEnable here.
    int ltv_state = sig_verify_result_array.getAt(i).getLTVState();
    if ((sig_verify_result_array.getAt(i).getSignatureState() & Signature.e_StateVerifyValid) ==
Signature.e_StateVerifyValid)
        ltv_verifier.addDSS(sig_verify_result_array.getAt(i));
}
}

// Add a time stamp signature as DTS and sign it. "saved_ltv_pdf_path" represents the newly saved signed PDF
file.
PDFPage pdf_page = pdf_doc.getPage(0);
// The new time stamp signature will have default filter name "Adobe.PPKLite" and default subfilter name
"ETSI.RFC3161".
Signature timestamp_signature = pdf_page.addSignature(new RectF(), "",
Signature.e_SignatureTypeTimeStamp, true);
Progressive sign_progressive = timestamp_signature.startSign("", empty_str.getBytes(),
Signature.e_DigestSHA256, saved_ltv_pdf_path, null, null);
if (sign_progressive.getRateOfProgress() != 100)
    sign_progressive.resume();

// Then use LTVVerfier to verify the new signed PDF file.
PDFDoc check_pdf_doc = new PDFDoc(saved_ltv_pdf_path);
check_pdf_doc.startLoad(null, false, null);
{
    // Use LTVVerfier to verify.
    LTVVerifier ltv_verifier = new LTVVerifier(pdf_doc, true, true, false, LTVVerifier.e_SignatureCreationTime);
    // Set verifying mode which is necessary.
    ltv_verifier.setVerifyMode(LTVVerifier.e_VerifyModeAcrobat);
    SignatureVerifyResultArray sig_verify_result_array = ltv_verifier.verify();
    for (long i = 0; i < sig_verify_result_array.getSize(); i++) {
        // ltv state would be e_LTVStateEnable here.
        int ltv_state = sig_verify_result_array.getAt(i).getLTVState();
        ... // User can get other information from SignatureVerifyResult.
    }
}

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

3.29 PAdES

From version 7.0, Foxit PDF SDK also supports PAdES (PDF Advanced Electronic Signature) which is the application for CAdES signature in the field of PDF. CAdES is a new standard for advanced digital

signature, its default subfilter is "ETSI.CAdES.detached". PAdES signature includes four levels: B-B, B-T, B-LT, and B-LTA.

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

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

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

3.30 PDF Action

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

Example:

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

```
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.URIAction;
import com.foxit.sdk.pdf.annots.Link;
...

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

// Add link annotation
Link link = new Link(page.addAnnot(Annot.e_Link, new RectF(350, 350, 380, 400)));
link.setHighlightingMode(Annot.e_HighlightingToggle);

// Add action for link annotation
```

```
Action action = Action.create(page.getDocument(), Action.e_TypeURI);
URIAction uriAction = new URIAction(action);
uriAction.setTrackPositionFlag(true);
uriAction.setURI("www.foxitsoftware.com");
link.setAction(uriAction);
// Appearance should be reset.
link.resetAppearanceStream();
...
```

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

```
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.GotoAction;
import com.foxit.sdk.pdf.annots.Link;
...

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

Link link = new Link(page.addAnnot(Annot.e_Link, new RectF(350, 350, 380, 400)));
link.setHighlightingMode(Annot.e_HighlightingToggle);

// Add action for link annotation
Action action = Action.create(page.getDocument(), Action.e_TypeGoto);
GotoAction gotoAction = new GotoAction(action);
Destination dest = Destination.createFitPage(doc, 0);
gotoAction.setDestination(dest);
...
```

3.31 JavaScript

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

JavaScript action is an action that causes a script to be compiled and executed by the JavaScript interpreter. Class `com.foxit.sdk.pdf.actions.JavaScriptAction` is derived from `Action` and offers functions to get/set JavaScript action data.

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

Example:

3.31.1 How to add JavaScript Action to Document

```
import com.foxit.sdk.pdf.actions.Action;
```

```
import com.foxit.sdk.pdf.actions.JavaScriptAction;
...

// Load Document doc.
...
JavaScriptAction javascript_action = new JavaScriptAction(Action.create(doc, Action.e_TypeJavaScript));
javascript_action.setScript("app.alert(\"Hello Foxit \");");
AdditionalAction aa = new AdditionalAction(doc);
aa.setAction(AdditionalAction.e_TriggerDocWillClose, javascript_action);
aa.doJSAction(AdditionalAction.e_TriggerDocWillClose);
...
```

3.31.2 How to add JavaScript Action to Annotation

```
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.JavaScriptAction;
...

// Load Document and get a widget annotation.
...
JavaScriptAction javascript_action = new JavaScriptAction(Action.create(page.getDocument(),
Action.e_TypeJavaScript));
javascript_action.setScript("app.alert(\"Hello Foxit \");");
AdditionalAction aa = new AdditionalAction(annot);
aa.setAction(AdditionalAction.e_TriggerAnnotMouseButtonPressed, javascript_action);
aa.doJSAction(AdditionalAction.e_TriggerAnnotMouseButtonPressed);
...
```

3.31.3 How to add JavaScript Action to FormField

```
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.JavaScriptAction;
import com.foxit.sdk.pdf.interform.Form;
import com.foxit.sdk.pdf.interform.Control;
import com.foxit.sdk.pdf.interform.Field;
...

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

// Add text field.
Control control = form.addControl(page, "Text Field0", Field.e_TypeTextField, new RectF(50, 600, 90, 640));
control.getField().setValue("3");
// Update text field's appearance.
control.getWidget().resetAppearanceStream();

Control control1 = form.addControl(page, "Text Field1", Field.e_TypeTextField, new RectF(100, 600, 140, 640));
control1.getField().setValue("23");
// Update text field's appearance.
control1.getWidget().resetAppearanceStream();

Control control2 = form.addControl(page, "Text Field2", Field.e_TypeTextField, new RectF(150, 600, 190, 640));
```



```
Field field2 = control2.getField();
JavaScriptAction javascript_action = new JavaScriptAction(Action.create(form.getDocument(),
Action.e_TypeJavaScript));
javascript_action.setScript("AFSimple_Calculate(\"SUM\", new Array (\"Text Field0\", \"Text Field1\");");
AdditionalAction aa = new AdditionalAction(field2);
aa.setAction(AdditionalAction.e_TriggerFieldRecalculateValue, javascript_action);
// Update text field's appearance.
control2.getWidget().resetAppearanceStream();
...
```

3.31.4 How to add a new annotation to PDF using JavaScript

```
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.JavaScriptAction;
...

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

JavaScriptAction javascript_action = new JavaScriptAction(Action.create(form.getDocument(),
Action.e_TypeJavaScript));
javascript_action.setScript("var annot = this.addAnnot({ page : 0, type : \"Square\", rect : [ 0, 0, 100, 100 ], name : \"UniqueID\", author : \"A. C. Robot\", contents : \"This section needs revision.\" });");
AdditionalAction aa = new AdditionalAction(field);
aa.setAction(AdditionalAction.e_TriggerAnnotCursorEnter, javascript_action);
aa.doJSAction(AdditionalAction.e_TriggerAnnotCursorEnter);
...
```

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

```
import com.foxit.sdk.pdf.actions.Action;
import com.foxit.sdk.pdf.actions.JavaScriptAction;
...

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

// Get properties of annotations.
JavaScriptAction javascript_action = new JavaScriptAction(Action.create(form.getDocument(),
Action.e_TypeJavaScript));
javascript_action.setScript("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);});");
AdditionalAction aa = new AdditionalAction(field);
aa.setAction(AdditionalAction.e_TriggerAnnotCursorEnter, javascript_action);
aa.doJSAction(AdditionalAction.e_TriggerAnnotCursorEnter);

// Set properties of annotations (only take strokeColor as an example).
JavaScriptAction javascript_action1 = new JavaScriptAction(Action.create(form.getDocument(),
Action.e_TypeJavaScript));
```

```
javascript_action1.setScript("var ann = this.getAnnot(0, \"UniqueID\"); if (ann != null) { ann.strokeColor =  
color.blue; }");  
AdditionalAction aa1 = new AdditionalAction(field1);  
aa1.setAction(AdditionalAction.e_TriggerAnnotCursorEnter, javascript_action1);  
aa1.doJSAction(AdditionalAction.e_TriggerAnnotCursorEnter);  
...
```

3.31.6 How to destroy annotation using JavaScript

```
import com.foxit.sdk.pdf.actions.Action;  
import com.foxit.sdk.pdf.actions.JavaScriptAction;  
...  
  
// Load Document and get form field, construct a Form object and a Filler object.  
...  
  
JavaScriptAction javascript_action = new JavaScriptAction(Action.create(form.getDocument(),  
Action.e_TypeJavaScript));  
javascript_action.setScript("var ann = this.getAnnot(0, \" UniqueID \"); if (ann != null) { ann.destroy(); } ");  
AdditionalAction aa = new AdditionalAction(field);  
aa.setAction(AdditionalAction.e_TriggerAnnotCursorEnter, javascript_action);  
aa.doJSAction(AdditionalAction.e_TriggerAnnotCursorEnter);  
...
```

3.32 Redaction

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

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

To do Redaction, you can use the following APIs:

- Call function [com.foxit.sdk.addon.Redaction.Redaction](#) to create a redaction module. If module "Redaction" is not defined in the license information which is used in function [common.Library.initialize](#), it means user has no right in using redaction related functions and this constructor will throw exception [com.foxit.sdk.common.Constants.e_ErrInvalidLicense](#) .
- Then call function [com.foxit.sdk.addon.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 `com.foxit.sdk.addon.Redaction.apply` to apply redaction in marked areas: remove the text or graphics under marked areas permanently.

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

Example:

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

```
import com.foxit.sdk.common.fxcrf.RectFArray;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.TextPage;
import com.foxit.sdk.pdf.TextSearch;
import com.foxit.sdk.addon.Redaction;
import com.foxit.sdk.pdf.annots.Redact;
...

// Assuming PDFDoc doc has been loaded.
...

Redaction redaction = new Redaction(doc);
PDFPage page = doc.getPage(0);
// Parse PDF page.
page.startParse(PDFPage.e_ParsePageNormal, null, false);
TextPage text_page = new TextPage(page, TextPage.e_ParseTextNormal);
TextSearch text_search = new TextSearch(text_page);
text_search.setPattern("PDF");
RectFArray matched_rect_array = new RectFArray();
while (text_search.findNext()) {
    RectFArray temp_rect_array = text_search.getMatchRects();
    for (int z=0; z<temp_rect_array.getSize(); z++)
        matched_rect_array.add(temp_rect_array.getAt(z));
}
if (matched_rect_array.getSize()>0) {
    Redact redact = redaction.markRedactAnnot(page, matched_rect_array);
    redact.resetAppearanceStream();
    doc.saveAs(output_path + "AboutFoxit_redacted_default.pdf", PDFDoc.e_SaveFlagNormal);

    // set border color to Green
    redact.setBorderColor((long)0x00FF00);
    // set fill color to Blue
    redact.setFillColor((long)0x0000FF);
    // set rollover fill color to Red
    redact.setApplyFillColor((long)0xFF0000);
    redact.resetAppearanceStream();
    doc.saveAs(output_path + "AboutFoxit_redacted_setColor.pdf", PDFDoc.e_SaveFlagNormal);

    redact.setOpacity((float)0.5);
    redact.resetAppearanceStream();
}
```

```
doc.saveAs(output_path+"AboutFoxit_redacted_setOpacity.pdf", PDFDoc.e_SaveFlagNormal);

redaction.apply();
}
...
```

3.33 Comparison

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

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

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

Example:

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

```
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.common.fxcr.PointF;
import com.foxit.sdk.common.fxcr.RectFArray;
import com.foxit.sdk.common.Image;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.addon.comparison.CompareResultInfo;
import com.foxit.sdk.addon.comparison.CompareResultInfoArray;
import com.foxit.sdk.addon.comparison.CompareResults;
import com.foxit.sdk.addon.comparison.Comparison;
import com.foxit.sdk.common.DateTime;
import com.foxit.sdk.pdf.annots.Annot;
import com.foxit.sdk.pdf.annots.Highlight;
import com.foxit.sdk.pdf.annots.Stamp;
import com.foxit.sdk.pdf.annots.QuadPoints;
import com.foxit.sdk.pdf.annots.QuadPointsArray;

...
PDFDoc base_doc = new PDFDoc("input_base_file");
error_code = base_doc.load(null);
if (error_code != e_ErrSuccess) {
    return;
}

PDFDoc compared_doc = new PDFDoc("input_compared_file");
error_code = compared_doc.load(null);
```

```

if (error_code != e_ErrSuccess) {
    return;
}

Comparison comparison = new Comparison(base_doc, compared_doc);

// Start comparing.
CompareResults result = comparison.doCompare(0, 0, Comparison.e_CompareTypeText);
CompareResultInfoArray oldInfo = result.getResults_base_doc();
CompareResultInfoArray newInfo = result.getResults_compared_doc();
long oldInfoSize = oldInfo.getSize();
long newInfoSize = newInfo.getSize();
PDFPage page = compared_doc.getPage(0);
for (int i=0; i<newInfoSize; i++)
{
    CompareResultInfo item = newInfo.getAt(i);
    int type = item.getType();
    if (type == CompareResultInfo.e_CompareResultTypeDeleteText)
    {
        String res_string;
        res_string = String.format("\'%s\'", item.getDiff_contents());
        CreateDeleteTextStamp(page, item.getRect_array(), 0xff0000, res_string, "Compare : Delete", "Text");
    }
    else if (type == CompareResultInfo.e_CompareResultTypeInsertText)
    {
        String res_string;
        res_string = String.format("\'%s\'", item.getDiff_contents());
        CreateDeleteText(page, item.getRect_array(), 0x0000ff, res_string, "Compare : Insert", "Text");
    }
    else if (type == CompareResultInfo.e_CompareResultTypeReplaceText)
    {
        String res_string;
        res_string = String.format("[Old]: \\'%s\'\\\'\\n[New]: \\'%s\'", oldInfo.getAt(i).getDiff_contents(),
item.getDiff_contents());
        CreateSquigglyRect(page, item.getRect_array(), 0xe7651a, res_string, "Compare : Replace", "Text");
    }
}

// Save the comparison result to a PDF file.
compared_doc.saveAs(output_path + "result.pdf", PDFDoc.e_SaveFlagNormal);

```

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

3.34 OCR

Optical Character Recognition, or OCR, is a software process that enables images or printed text to be translated into machine-readable text. OCR is most commonly used when scanning paper

documents to create electronic copies, but can also be performed on existing electronic documents (e.g. PDF).

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

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

3.34.1 System requirements

Platform: Windows, Linux (x64)

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

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

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

3.34.2 Trial limit for SDK OCR add-on module

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

- 1) Allow 30 consecutive natural days to evaluate SDK from the first time of OCR Engine initialization.
- 2) Allow up to 5000 pages to be converted using OCR from the first time of OCR Engine initialization.
- 3) Trail watermarks will be generated on the PDF pages. This limit is used for all of the SDK modules.

3.34.3 OCR resource files

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

For Windows:

After getting the package for Windows, 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.

For Linux x64:

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

- **Data:** Data and resource files for following languages:
Chinese-Simplified, Chinese-Traditional, Japanese, Korean, 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.
- **Bin:** Library files for Linux x64.

3.34.4 How to run the OCR demo

Foxit PDF SDK for Java API (Windows and Linux x64) provides an OCR demo located in the "`\examples\simple_demo\ocr`" folder to show you how to use Foxit PDF SDK to do OCR for a PDF page or a PDF document.

3.34.4.1 Build an OCR resource directory

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

For Windows:

To build an OCR resource directory on Windows, 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.

For Linux x64:

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

- 1) Create a new folder to add the resources. For example, "/root/Desktop/ocr_resources".
- 2) Copy the whole folders of "**Data**", "**Bin**" under the "ocr_addon_linux" to "/root/Desktop/ocr_resources".

Then, the OCR resource files path is set to "**/root/Desktop/ocr_resources/Bin**".

Note: Before loading the resource files, please set the environment variable for loading the library, please execute: `export LD_LIBRARY_PATH=/root/Desktop/ocr_resources/Bin`.

3.34.4.2 Configure the demo

After building the OCR resource directory, configure the demo in the "`examples\simple_demo\ocr\ocr.java`" file. Following will configure the demo in "`ocr.java`" file on Windows for example. For Linux x64 platform, do the similar configuration with Windows.

Specify the OCR resource directory

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

```
45 // "ocr_resource_path" is the path of ocr resources. Please refer to Developer Guide for more details.
46 String ocr_resource_path = "D:/ocr_resources";
47
48 if (ocr_resource_path == "") {
49     System.out.println("ocr_resource_path is still empty. Please set it with a valid path to OCR resource path.");
50     return;
51 }
52
53 // Initialize OCR engine.
54 error_code = OCREngine.initialize(ocr_resource_path);
```

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 **OCREngine.setLanguages** method and is set to "English" by default.

```
71 // Set languages.
72 OCREngine.setLanguages("English");
```

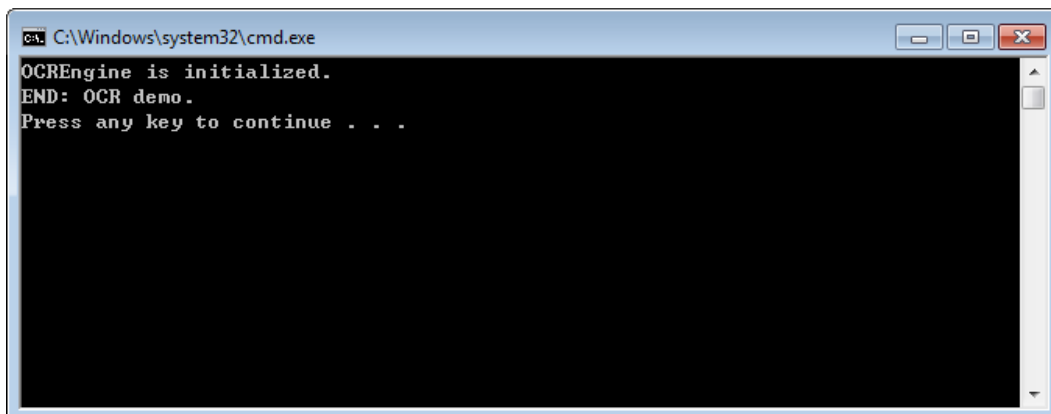
(Optional) Set log for OCREngine

If you want to print the entire log of the OCR Engine, please uncomment the **OCREngine.setLogFile** method as below:

```
68 // Set log for OCREngine. (This can be opened to set log file if necessary)
69 OCREngine.setLogFile(output_path+"ocr.log");
```

3.34.4.3 Run the demo

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



```
C:\Windows\system32\cmd.exe
OCREngine is initialized.
END: OCR demo.
Press any key to continue . . .
```

The demo will OCR the default document

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

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

3.35 Compliance

PDF Compliance

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

PDF/A Compliance

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

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

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

Note: For Mac Java, the **pdfEngineLight.dylib** file should be copied from the **var\\legacy** folder and then replaced in the root directory of the engine. Additionally, it does not support multithreading.

3.35.1 System requirements

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

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

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

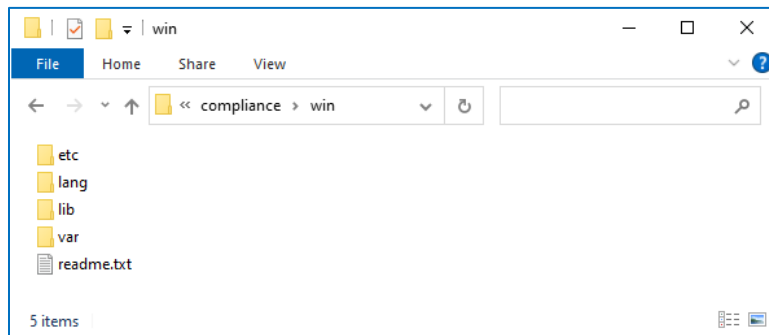
SDK Version: Foxit PDF SDK (C++, Java, C#, Objective-C) 6.4 or higher (for PDF Compliance, it requires Foxit PDF SDK 7.1 or higher); Foxit PDF SDK (C) 7.4 or higher; Foxit PDF SDK (Python) 8.3 or higher

3.35.2 Compliance resource files

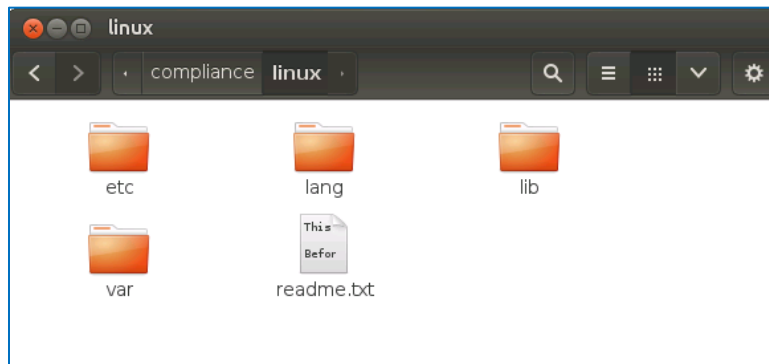
Please contact Foxit support team or sales team to get the Compliance resource files package.

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

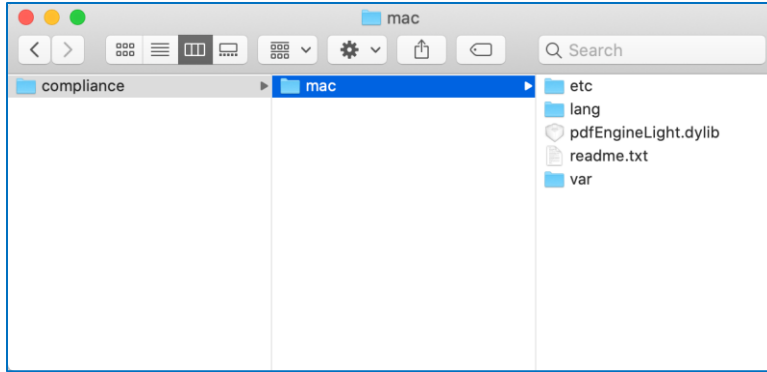
For **Windows**:



For **Linux**:



For **Mac**:



3.35.3 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.

3.35.3.1 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 **ComplianceEngine.initialize** to initialize compliance engine.

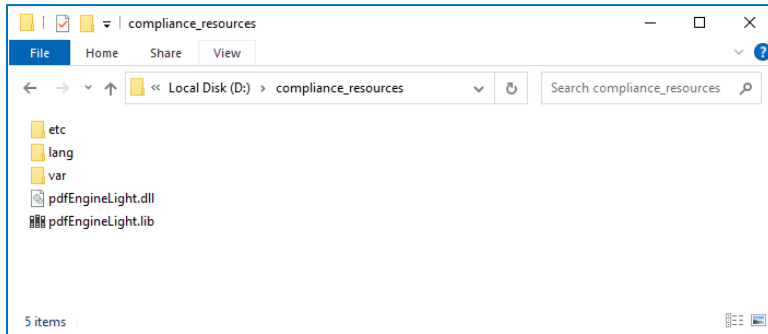
Starting from version 10.0, the compliance resource files provide default thread-safety. For multithreading, the API **ComplianceEngine.initializeThreadContext** should be called first for a new thread before using any other methods in the compliance add-on module.

Windows

To build a compliance resource directory on Windows, 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:

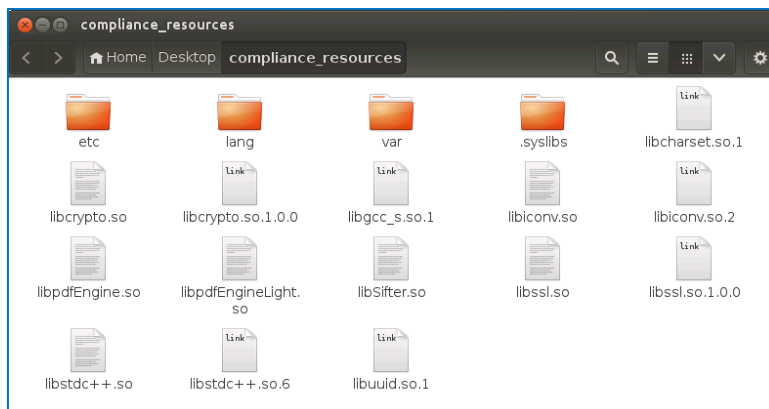


Linux

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

- 1) Create a new folder to add the resources. For example, `"/root/Desktop/compliance_resources"`.
- 2) Copy the whole folders of "**ect**", "**lang**", "**var**" under the "compliance/linux" to `"/root/Desktop/compliance_resources"`.
- 3) Add the appropriate library resource based on the platform architecture.
 - For **linux32**, copy **all the files** under "compliance/linux/lib/x86" folder to `"/root/Desktop/compliance_resources"`.
 - For **linux64**, copy **all the files** under "compliance/linux/lib/x64" folder to `"/root/Desktop/compliance_resources"`.

For example, use **linux32** platform architecture, then the compliance resource directory should be as follows:



Note: For Linux platform, you should put the compliance resource directory into the search path for system shared library before running the demo, otherwise **ComplianceEngine.initialize** will fail.

For example, you can use the command (`export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/root/Desktop/compliance_resources`) to temporarily add the compliance resource directory to `LD_LIBRARY_PATH`.

Mac

For Mac platform, you can directly use the "compliance/mac" resource folder as the compliance resource directory.

3.35.3.2 Configure the demo

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

This section takes **Windows** as an example to show you how to configure the demo in the "compliance.java" file. For Linux and Mac, do the same configuration with Windows.

Specify the compliance resource directory

In the "compliance.java" file, add the compliance resource directory as follows, which will be used to initialize the compliance engine.

```
125     try {
126         // "compliance_resource_folder_path" is the path of compliance resource folder. Please refer to Developer Guide for more details.
127         String compliance_resource_folder_path = "D:/compliance_resources";
128         // If you use an authorization key for Foxit PDF SDK, please set a valid unlock code string to compliance_engine_unlockcode for ComplianceEngine.
129         // If you use a trial key for Foxit PDF SDK, just keep compliance_engine_unlockcode as an empty string.
130         String compliance_engine_unlockcode = "";
131
132         if (compliance_resource_folder_path.length() < 1) {
133             System.out.println("compliance_resource_folder_path is still empty. Please set it with a valid path to compliance resource folder path.");
134             return ;
135         }
136         // Initialize compliance engine.
137         error_code = ComplianceEngine.initialize(compliance_resource_folder_path, compliance_engine_unlockcode);
```

Note:

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

(Optional) Set language for compliance engine

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, uncomment the **ComplianceEngine.setLanguage** method, and set the language to "Chinese-Simplified".

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

(Optional) Set a temp folder for compliance engine

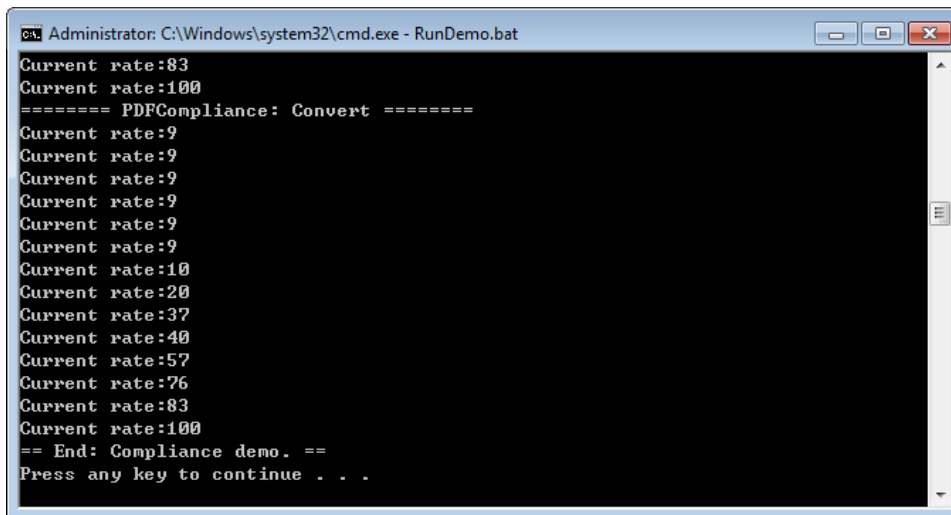
ComplianceEngine.setTempFolderPath function is used to set a temp folder to store several files for proper processing (e.g verifying or converting). If no custom temp folder is set by this function, the default temp folder in system will be used.

For example, uncomment the **ComplianceEngine.setTempFolderPath** method, and set the path to "D:/compliance_temp" (should be a valid path).

```
151 // Set custom temp folder path for ComplianceEngine.
152 ComplianceEngine.setTempFolderPath("D:/compliance_temp");
```

3.35.3.3 Run the demo

Locate to "\\examples\\simple_demo\\compliance", and run "RunDemo.bat", then the console will print the following by default:



```
Administrator: C:\Windows\system32\cmd.exe - RunDemo.bat
Current rate:83
Current rate:100
===== PDFCompliance: Convert =====
Current rate:9
Current rate:9
Current rate:9
Current rate:9
Current rate:9
Current rate:9
Current rate:10
Current rate:20
Current rate:37
Current rate:40
Current rate:57
Current rate:76
Current rate:83
Current rate:100
== End: Compliance demo. ==
Press any key to continue . . .
```

The 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.

3.36 Optimization

Optimization feature can reduce the size of PDF files to save disk space and make files easier to send and store, through compressing images, deleting redundant data, discarding useless user data and so on. From version 7.0, optimization module provides functions to compress the color/grayscale/monochrome images in PDF files to reduce the size of the PDF files.

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

Example:

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

```
import com.foxit.sdk.common.Progressive;
import com.foxit.sdk.common.Constants;
import com.foxit.sdk.common.fxcrp.PauseCallback;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.addon.*;
import com.foxit.sdk.addon.optimization.*;
import com.foxit.sdk.addon.optimization.Optimizer;

import java.io.File;
import static com.foxit.sdk.common.Constants.e_ErrSuccess;
import static com.foxit.sdk.pdf.PDFDoc.e_SaveFlagRemoveRedundantObjects;

PDFDoc doc = new PDFDoc("input_pdf_file");
error_code = doc.load(null);
if (error_code != e_ErrSuccess) {
    System.out.println(" Error: " + error_code);
    return;
}
PauseUtil pause = new PauseUtil();
OptimizerSettings settings = new OptimizerSettings();
System.out.println("Optimized Start.");
Progressive progressive = Optimizer.optimize(doc, settings, pause);
int state = Progressive.e_ToBeContinued;
while (state == Progressive.e_ToBeContinued) {
    state = progressive.resume();
    int rate = progressive.getRateOfProgress();
    System.out.println("Optimize progress percent: " + rate + "%");
}
if (state == Progressive.e_Finished)
```



```
{  
    doc.saveAs("ImageCompression_Optimized.pdf", e_SaveFlagRemoveRedundantObjects);  
}  
System.out.println("Optimized Finish.");
```

3.37 HTML to PDF Conversion

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

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

From version 7.6, Foxit PDF SDK supports to convert HTML to PDF on Linux platform. But for HTML to PDF engine (Linux), the version of libnss should be 3.22.

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

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

3.37.1 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

3.37.2 HTML to PDF engine files

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

After getting the package, extract it to a desired directory (for example, extract the package to a directory: "**htmltopdf/win**" for Windows, "**htmltopdf/linux**" for Linux, and "**htmltopdf/mac**" for Mac).

3.37.3 How to run the html2pdf demo

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

3.37.3.1 Prepare a HTML2PDF engine directory

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

3.37.3.2 Configure the demo

For html2pdf demo, you can configure the demo in the "\\examples\\simple_demo\\html2pdf**html2pdf.java**" file, or you can configure the demo with parameters directly in a command prompt. Following will configure the demo in "html2pdf.java" file on Windows for example. For Linux and Mac platform, do the same configuration with Windows.

Specify the html2pdf engine directory

In the "html2pdf.java" 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.  
private static String engine_path = "D:/htmltopdf/win/fxhtml2pdf.exe"; // or engine_path = "D:/htmltopdf/win/fxhtml2pdf";
```

(Optional) Specify cookies file path

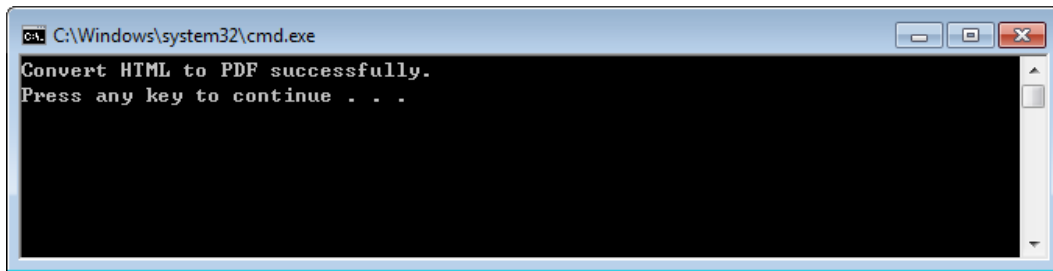
Add the path of the cookies file exported from the web pages that you want to convert. For example,

```
// "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.  
private static String cookies_path = "D:/cookies.txt";
```

3.37.3.3 Run the demo

Run the demo without parameters

Locate to "\\examples\\simple_demo\\html2pdf", and run "RunDemo.bat", then the console will print the following by default:

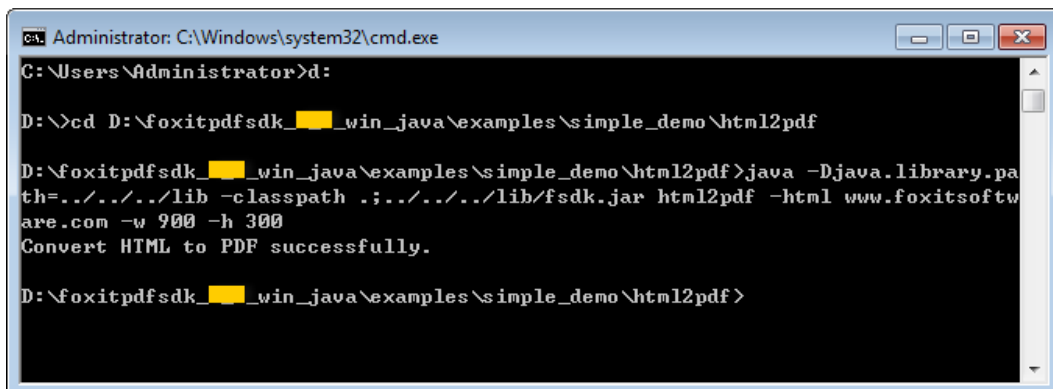


Run the demo with parameters

Locate to "\examples\simple_demo\html2pdf", and run "RunDemo.bat" at first.

Then, open a command prompt, navigate to "\examples\simple_demo\html2pdf", type "java -Djava.library.path=../../lib -classpath ../../lib/fsdk.jar html2pdf --help" 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.
-engine	The path of the engine file "fxhtml2pdf.exe".
-w	The page width of the output PDF file in points.
-h	The page height of the output PDF file in points.
-r	The page rotation for the output PDF file. <ul style="list-style-type: none">• 0 : 0 degree.• 1 : 90 degree.• 2 : 180 degree.• 3 : 270 degree.
-ml	The left margin of the pages for the output PDF file.
-mr	The right margin of the pages for the output PDF file.
-mt	The top margin of the pages for the output PDF file.
-mb	The bottom margin of the pages for the output PDF file.
-mode	The page mode for the output PDF file. <ul style="list-style-type: none">• 0 : Single page mode.• 1 : Multiple pages mode.
-scale	The scaling mode. <ul style="list-style-type: none">• 0 : No need to scale pages.• 1 : Scale pages.• 2 : Enlarge page.
-link	Whether to convert links. <ul style="list-style-type: none">• 'yes' : Convert links.• 'no' : No need to convert links.

Parameters	Description
-tag	Whether to generate tag. <ul style="list-style-type: none">• 'yes' : Generate tag.• 'no' : No need to generate tag.
-bookmarks	Whether to generate bookmarks. <ul style="list-style-type: none">• 'yes' : Generate bookmarks .• 'no' : No need to generate bookmarks.
-print_background	Whether to print background. <ul style="list-style-type: none">• 'yes' : Print bookmarks .• 'no' : No need to print bookmarks.
-optimize_tag	Whether to optimize tag tree. <ul style="list-style-type: none">• 'yes' : Optimize tag tree .• 'no' : No need to optimize tag tree.
-media	The media style. <ul style="list-style-type: none">• 0 : Screen media style.• 1 : Print media style.
-encoding	The HTML encoding format. <ul style="list-style-type: none">• 0 : Auto encoding .• 1-73 : Other encodings.
-render_images	Whether to render images. <ul style="list-style-type: none">• 'yes' : Render images.• 'no' : No need to render images.
-remove_underline_for_link	Whether to remove underline for link. <ul style="list-style-type: none">• 'yes' : Remove underline for link.• 'no' : No need to remove underline for link.
-headerfooter	Whether to generate headerfooter. <ul style="list-style-type: none">• 'yes' : Generate headerfooter.• 'no' : No need to generate headerfooter.
-headerfooter_title	The headerfooter title.
-headerfooter_url	The headerfooter url.
-bookmark_root_name	The bookmark root name.
-resize_objects	Whether to enable the JavaScripts related resizing of the objects during rendering process. <ul style="list-style-type: none">• 'yes' : Enable.• 'no' : Disable.

Parameters	Description
-cookies	The path of the cookies file exported from a URL that you want to convert.
-timeout	The timeout of loading webpages.

3.37.4 How to work with Html2PDF API

```
import com.foxit.sdk.addon.conversion.HTML2PDFSettingData;

com.foxit.sdk.addon.conversion.HTML2PDFSettingData pdf_setting_data =
new com.foxit.sdk.addon.conversion.HTML2PDFSettingData();
pdf_setting_data.setIs_convert_link(true);
pdf_setting_data.setIs_generate_tag(true);
pdf_setting_data.setTo_generate_bookmarks(true);
pdf_setting_data.setRotate_degrees(0);
pdf_setting_data.setPage_width(900);
pdf_setting_data.setPage_height(640);
pdf_setting_data.setPage_mode(HTML2PDFSettingData.e_PageModeSinglePage);
pdf_setting_data.setScaling_mode(HTML2PDFSettingData.e_ScalingModeScale);
pdf_setting_data.setTo_print_background(true);
pdf_setting_data.setTo_optimize_tag_tree(false);
pdf_setting_data.setMedia_style(HTML2PDFSettingData.e_MediaStyleScreen);
...

com.foxit.sdk.addon.conversion.Convert.fromHTML(url_or_html, engine_path, cookies_path, pdf_setting_data,
output_pdf_path, timeout);
```

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

1. Defines a [FileRead](#) class inherited from [FileReaderCallback](#) used to get html data from stream or memory. And defines a [FileWriter](#) class inherited from [FileWriterCallback](#) used to do file writing. For the implementations of [FileRead](#) and [FileWriter](#) classes, please refer to the **html2pdf** demo in the "examples\simple_demo\html2pdf" folder.
2. Get html data from stream and set resources related to source html.
3. Call the [com.foxit.sdk.addon.conversion.Convert.fromHTML](#) function to convert it to a PDF file.

```
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.addon.conversion.HTML2PDFSettingData;
import com.foxit.sdk.common.fxcr.FileReaderCallback;
import com.foxit.sdk.common.fxcr.FileWriterCallback;
import com.foxit.sdk.addon.conversion.HTML2PDFRelatedResourceArray;
import com.foxit.sdk.addon.conversion.HTML2PDFRelatedResource;

com.foxit.sdk.addon.conversion.HTML2PDFSettingData pdf_setting_data=new
com.foxit.sdk.addon.conversion.HTML2PDFSettingData();
pdf_setting_data.setPage_height(650);
```

```
pdf_setting_data.setPage_width(950);
pdf_setting_data.setIs_to_page_scale(false);
pdf_setting_data.setPage_margin(new RectF(18, 18, 18, 18));
pdf_setting_data.setIs_convert_link(true);
pdf_setting_data.setRotate_degrees(0);
pdf_setting_data.setIs_generate_tag(true);
pdf_setting_data.setPage_mode(HTML2PDFSettingData.e_PageModeSinglePage);
pdf_setting_data.setScaling_mode(HTML2PDFSettingData.e_ScalingModeScale);
pdf_setting_data.setTo_generate_bookmarks(true);
pdf_setting_data.setEncoding_format(0);
pdf_setting_data.setTo_render_images(true);
pdf_setting_data.setTo_remove_underline_for_link(false);
pdf_setting_data.setTo_set_headerfooter(false);
pdf_setting_data.setHeaderfooter_title("");
pdf_setting_data.setHeaderfooter_url("");
pdf_setting_data.setBookmark_root_name("abcde");
pdf_setting_data.setTo_resize_objects(true);
pdf_setting_data.setTo_print_background(false);
pdf_setting_data.setTo_optimize_tag_tree(false);
pdf_setting_data.setMedia_style(0);
pdf_setting_data.setTo_load_active_content();
String output_path = output_directory + "html2pdf_filestream_result.pdf";
FoxitFileWriter filewrite = new FoxitFileWriter();
filewrite.LoadFile(output_path);
// "htmlfile" is the path of the html file to be loaded. For example: "C:/aaa.html". The method of "FromHTML"
will load this file as a stream.
String htmlfile = "";
FileRead filereader = new FileRead();
filereader.loadFile(htmlfile,false);

// "htmlfilepng" is the path of the png resource file to be loaded. For example: "C:/aaa.png". set "htmlfilepng" in
the related_resource_file of HTML2PDFRelatedResource.
String htmlfilepng = "";
FileRead filereader1 = new FileRead();
filereader1.loadFile(htmlfilepng,false);
// "relativefilepath" is the resource file's relative path. For example: "./aaa.png".
String relativefilepath = "";
com.foxit.sdk.addon.conversion.HTML2PDFRelatedResourceArray html2PDFRelatedResourceArray = new
com.foxit.sdk.addon.conversion.HTML2PDFRelatedResourceArray();
com.foxit.sdk.addon.conversion.HTML2PDFRelatedResource html2PDFRelatedResource = new
com.foxit.sdk.addon.conversion.HTML2PDFRelatedResource(filereader1,relativefilepath);

html2PDFRelatedResourceArray.add(html2PDFRelatedResource);
com.foxit.sdk.addon.conversion.Convert.fromHTML(filereader, html2PDFRelatedResourceArray, engine_path,
null,
pdf_setting_data, filewrite, 30);
System.out.println("Convert HTML to PDF successfully by filestream!");
```

3.38 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 convert PowerPoint documents into PDF files on Windows platform.

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

For using this feature, please note that:

- Make sure that Microsoft Office 2007 version or higher is already installed on your Windows system.
- Before converting Excel to PDF, make sure that the default Microsoft virtual printer is already set on your Windows system.
- For Linux x86/x64, make sure that LibreOffice is already installed on your Linux system.

Note: When using LibreOffice 7.0 or a higher version, if you encounter an error like "An unknown error has occurred", you can try to set an environment variable before running the program as follows:

```
"export URE_BOOTSTRAP=vnd.sun.star.pathname:/opt/libreoffice7.x/program/fundamentalrc"
```

Where, 'x' represents the LibreOffice version.

- For Linux armv8, make sure that kingsoft wps-office is already installed on your Linux system.

3.38.1 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:

Note:

- For Linux x86/x64, the parameter "`engine_path`" in the following sample codes represents the path of LibreOffice engine. To get the installed path of LibreOffice, you can input the command "**locate soffice.bin**" in a terminal, then the path will be shown, for example, `/usr/lib/libreoffice/program/soffice.bin`. Then the value of "`engine_path`" parameter is set to `/usr/lib/libreoffice/program`.
- For Linux armv8, the parameter "`engine_path`" in the following sample codes represents the path of WPS engine (`librpcwpsapi.so`). Find the directory where "`librpcwpsapi.so`" is located. For example, the directory is `/opt/kingsoft/wps-office/office6`. Then the value of "`engine_path`" parameter is set to `/opt/kingsoft/wps-office/office6`.

3.38.2 How to convert Word to PDF

Windows

```
import com.foxit.sdk.addon.conversion.Word2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

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

String word_file_path = "test.doc";
String saved_pdf_path = "saved.pdf";

// Use default Word2PDFSettingData values.
Word2PDFSettingData word_convert_setting_data = new Word2PDFSettingData();
Convert.fromWord(word_file_path, "", saved_pdf_path, word_convert_setting_data);
```

Linux (x86 and x64)

```
import com.foxit.sdk.addon.conversion.Word2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

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

String word_file_path = "test.doc";
String saved_pdf_path = "saved.pdf";

// Use default Word2PDFSettingData values.
Word2PDFSettingData word_convert_setting_data = new Word2PDFSettingData();
Convert.fromWord(word_file_path, "", saved_pdf_path, engine_path, word_convert_setting_data);
```

3.38.3 How to convert Excel to PDF

Windows

```
import com.foxit.sdk.addon.conversion.Word2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;
```

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

String excel_file_path = "test.xls";
String saved_pdf_path = "saved.pdf";

// Use default Excel2PDFSettingData values.
Excel2PDFSettingData excel_convert_setting_data = new Excel2PDFSettingData();
Convert.fromExcel(excel_file_path, "", saved_pdf_path, excel_convert_setting_data);
```

Linux (x86 and x64)

```
import com.foxit.sdk.addon.conversion.Word2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

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

String excel_file_path = "test.xls";
String saved_pdf_path = "saved.pdf";

// Use default Excel2PDFSettingData values.
Excel2PDFSettingData excel_convert_setting_data = new Excel2PDFSettingData();
Convert.fromExcel(excel_file_path, "", saved_pdf_path, engine_path, excel_convert_setting_data);
```

3.38.4 How to convert PowerPoint to PDF

Windows

```
import com.foxit.sdk.addon.conversion.PowerPoint2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

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

String ppt_file_path = "test.ppt";
String saved_pdf_path = "saved.pdf";

// Use default PowerPoint2PDFSettingData values.
PowerPoint2PDFSettingData ppt_convert_setting_data = new PowerPoint2PDFSettingData();
Convert.fromPowerPoint(ppt_file_path, "", saved_pdf_path, ppt_convert_setting_data);
```

Linux (x86 and x64)

```
import com.foxit.sdk.addon.conversion.PowerPoint2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

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

String ppt_file_path = "test.ppt";
String saved_pdf_path = "saved.pdf";

// Use default PowerPoint2PDFSettingData values.
PowerPoint2PDFSettingData ppt_convert_setting_data = new PowerPoint2PDFSettingData();
```

```
Convert.fromPowerPoint(ppt_file_path, "", saved_pdf_path, engine_path, ppt_convert_setting_data);
```

3.39 Output Preview

From version 7.4, Foxit PDF SDK supports output preview feature which can preview color separations and test different color profiles.

3.39.1 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

3.39.2 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.  
String default_icc_folder_path = "E:/foxitpdfsdk_X_X_win_java/res/icc_profile";
```

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

3.39.3 How to do output preview using Foxit PDF SDK

```
import com.foxit.sdk.common.Library;  
import com.foxit.sdk.common.Renderer;  
import com.foxit.sdk.common.fxcr.Matrix2D;  
import com.foxit.sdk.pdf.OutputPreview;  
  
// Make sure that SDK has already been initialized successfully.  
  
// Set folder path which contains default icc profile files.  
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.  
  
OutputPreview output_preview = new OutputPreview(pdf_doc);  
String simulation_icc_file_path = input_path + "icc_profile/USWebCoatedSWOP.icc";  
output_preview.setSimulationProfile(simulation_icc_file_path);  
output_preview.setShowType(OutputPreview.e_ShowAll);  
ArrayList<String> process_plates = output_preview.getPlates(OutputPreview.e_ColorantTypeProcess);
```

```
ArrayList<String> spot_plates = output_preview.getPlates(OutputPreview.e_ColorantTypeSpot);

// Set check status of process plate to be true, if there's any process plate.
for (int i = 0; i < (int)process_plates.size(); i++) {
    output_preview.setCheckStatus(process_plates.get(i), true);
}

// Set check status of spot plate to be true, if there's any spot plate.
for (int i = 0; i < (int)spot_plates.size(); i++) {
    output_preview.setCheckStatus(spot_plates.get(i), true);
}

Bitmap preview_bitmap = output_preview.generatePreviewBitmap(pdf_page, display_matrix, renderer);
```

3.40 Combination

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

3.40.1 How to combine several PDF files into one PDF file

```
import com.foxit.sdk.common.Progressive;
import com.foxit.sdk.pdf.CombineDocumentInfo;
import com.foxit.sdk.pdf.CombineDocumentInfoArray;
import com.foxit.sdk.pdf.Combination;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionBookmark;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionAcroformRename;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionStructTree;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionOutputIntents;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionOCProperties;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionMarkInfos;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionPageLabels;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionNames;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionObjectStream;
import static com.foxit.sdk.pdf.Combination.e_CombineDocsOptionDuplicateStream;

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

CombineDocumentInfoArray info_array = new CombineDocumentInfoArray();
info_array.add(new CombineDocumentInfo(input_path + "AboutFoxit.pdf", ""));
info_array.add(new CombineDocumentInfo(input_path + "Annot_all.pdf", ""));
info_array.add(new CombineDocumentInfo(input_path + "SamplePDF.pdf", ""));

String savepath = output_path + "Test_Combined.pdf";
int option = (int)(e_CombineDocsOptionBookmark | e_CombineDocsOptionAcroformRename |
    e_CombineDocsOptionStructTree | e_CombineDocsOptionOutputIntents |
    e_CombineDocsOptionOCProperties | e_CombineDocsOptionMarkInfos |
    e_CombineDocsOptionPageLabels | e_CombineDocsOptionNames |
    e_CombineDocsOptionObjectStream | e_CombineDocsOptionDuplicateStream);

Progressive progress = Combination.startCombineDocuments(savepath, info_array, option, null);
```

```
int progress_state = Progressive.e_ToBeContinued;
while (Progressive.e_ToBeContinued == progress_state)
{
    progress_state = progress.resume();
}
```

3.41 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.

3.41.1 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:

3.41.2 How to create a new and blank PDF portfolio

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.Portfolio;
import com.foxit.sdk.common.Constants;

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

Portfolio new_portfolio = Portfolio.createPortfolio();

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

// Get portfolio PDF document object.
PDFDoc portfolio_pdf_doc = new_portfolio.getPortfolioPDFDoc();
```

3.41.3 How to create a Portfolio object from a PDF portfolio

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.Portfolio;
import com.foxit.sdk.common.Constants;

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

PDFDoc pdf_doc = new PDFDoc("portfolio.pdf");
int error_code = portfolio_pdf_doc.load(null);
```

```
if (Constants.e_ErrSuccess == error_code) {
    if (portfolio_pdf_doc.isPortfolio()) {
        Portfolio existed_portfolio = Portfolio.createPortfolio(portfolio_pdf_doc);
    }
}
```

3.41.4 How to get portfolio nodes

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.Portfolio;
import com.foxit.sdk.pdf.SchemaField;
import com.foxit.sdk.pdf.SchemaFieldArray;
import com.foxit.sdk.pdf.PortfolioNode;
import com.foxit.sdk.pdf.PortfolioNodeArray;
import com.foxit.sdk.pdf.PortfolioFileNode;
import com.foxit.sdk.pdf.PortfolioFolderNode;
import com.foxit.sdk.pdf.FileSpec;
import com.foxit.sdk.common.Constants;

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

// Portfolio object has been created, assuming it is named "portfolio".
...

PortfolioNode root_node = portfolio.getRootNode();
PortfolioFolderNode root_folder = new PortfolioFolderNode(root_node);
PortfolioNodeArray sub_nodes = root_folder.getSortedSubNodes();
for (int index = 0; index < sub_nodes.getSize(); index++) {
    PortfolioNode node = sub_nodes.getAt(index);
    switch(node.getNodeType()) {
        case PortfolioNode::e_TypeFolder: {
            PortfolioFolderNode folder_node = new PortfolioFolderNode(node);

            // Use PortfolioFolderNode's methods to get some properties.
            ...

            PortfolioNodeArray sub_nodes_2 = folder_node.getSortedSubNodes();
            ...
            break;
        }
        case PortfolioNode::e_TypeFile: {
            PortfolioFileNode file_node = new PortfolioFileNode(node);
            // Get file specification object from this file node, and then get/set information from/to this file specification
            object.
            FileSpec file_spec = file_node.getFileSpec();
            ...
            break;
        }
    }
}
```

3.41.5 How to add file node or folder node

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.Portfolio;
import com.foxit.sdk.pdf.SchemaField;
import com.foxit.sdk.pdf.SchemaFieldArray;
import com.foxit.sdk.pdf.PortfolioNode;
import com.foxit.sdk.pdf.PortfolioNodeArray;
import com.foxit.sdk.pdf.PortfolioFileNode;
import com.foxit.sdk.pdf.PortfolioFolderNode;
import com.foxit.sdk.pdf.FileSpec;
import com.foxit.sdk.common.Constants;

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

// Add file from path.
String path_to_a_file = "directory/Sample.txt";
PortfolioFileNode new_file_node_1 = root_folder.addFile(path_to_a_file);

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

// Add file from MyStreamCallback which is inherited from StreamCallback and implemented by user.
MyStreamCallback my_stream_callback = new MyStreamCallback();
PortfolioFileNode new_file_node_2 = root_folder.addFile(my_stream_callback, "file_name");

// 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".
...

PortfolioFileNode new_file_node_3 = root_folder.addPDFDoc(test_pdf_doc, "pdf_file_name");

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

// Add a sub folder in root_folder.
PortfolioFolderNode new_sub_foldernode = root_folder.addSubFolder("Sub Folder-1");

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

3.41.6 How to remove a node

```
import com.foxit.sdk.common.Library;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.Portfolio;
import com.foxit.sdk.pdf.SchemaField;
```

```
import com.foxit.sdk.pdf.SchemaFieldArray;
import com.foxit.sdk.pdf.PortfolioNode;
import com.foxit.sdk.pdf.PortfolioNodeArray;
import com.foxit.sdk.pdf.PortfolioFileNode;
import com.foxit.sdk.pdf.PortfolioFolderNode;
import com.foxit.sdk.pdf.FileSpec;
import com.foxit.sdk.common.Constants;

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

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

3.42 Table Maker

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

3.42.1 System requirements

Platform: Windows, Mac, Linux

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

3.42.2 How to add table to a PDF document

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

```
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.common.fxcr.FloatArray;
import com.foxit.sdk.common.Image;

import com.foxit.sdk.pdf.annots.RichTextStyle;
import com.foxit.sdk.addon.tablegenerator.TableBorderInfo;
import com.foxit.sdk.addon.tablegenerator.TableData;
import com.foxit.sdk.addon.tablegenerator.TableCellData;
import com.foxit.sdk.addon.tablegenerator.TableCellDataArray;
import com.foxit.sdk.addon.tablegenerator.TableCellDataColArray;
import com.foxit.sdk.addon.tablegenerator.TableCellIndexArray;
import com.foxit.sdk.addon.tablegenerator.TableGenerator;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
```



```
int index = 0;
TableCellDataArray cell_array = new TableCellDataArray();

for (int row = 0; row < 4; row++) {
    TableCellDataColArray col_array = new TableCellDataColArray();
    for (int col = 0; col < 3; col++) {
        String cell_text = GetTableCellText(index);
        RichTextStyle style = GetTableCellStyle(index++);
        TableCellData cell_data = new TableCellData(style, cell_text, new Image(), new RectF());
        col_array.add(cell_data);
    }
    cell_array.add(col_array);
}

float page_width = pdf_page.getWidth();
float page_height = pdf_page.getHeight();
RectF rect = new RectF(100, 550, page_width - 100, page_height - 100);
TableBorderInfo outsideborderinfoleft = new TableBorderInfo();
outsideborderinfoleft.setLine_width(1);
TableBorderInfo outsideborderinfoforight = new TableBorderInfo();
outsideborderinfoforight.setLine_width(1);
TableBorderInfo outsideborderinfofotop = new TableBorderInfo();
outsideborderinfofotop.setLine_width(1);
TableBorderInfo outsideborderinfofobottom = new TableBorderInfo();
outsideborderinfofobottom.setLine_width(1);

TableBorderInfo insideborderinfo_row = new TableBorderInfo();
insideborderinfo_row.setLine_width(1);
TableBorderInfo insideborderinfo_col = new TableBorderInfo();
insideborderinfo_col.setLine_width(1);
TableData data = new TableData(rect, 4, 3, outsideborderinfoleft, outsideborderinfoforight,
    outsideborderinfofotop, outsideborderinfofobottom, insideborderinfo_row, insideborderinfo_col, new
    TableCellIndexArray(), new FloatArray(), new FloatArray());
TableGenerator.addTableToPage(pdf_page, data, cell_array);
```

3.43 Accessibility

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

3.43.1 System requirements

Platform: Windows, Mac, Linux

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

3.43.2 How to tag a PDF document

```
import com.foxit.sdk.common.Progressive;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.addon.accessibility.*;
import com.foxit.sdk.common.fxcrf.RectF;
....
PDFDoc pdfDoc= new PDFDoc(input_file);
pdfDoc.load(null);
TaggedPDF taggedpdf = new TaggedPDF(pdfDoc);
Progressive progressive = taggedpdf.startTagDocument(null);
int progressState = Progressive.e_ToBeContinued;
while (Progressive.e_ToBeContinued == progressState)
    progressState = progressive.resume();

pdfDoc.saveAs(output_file_path, 0);
....
```

3.44 PDF to Office Conversion

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

3.44.1 System requirements

Platform: Windows, Linux

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

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

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

3.44.2 PDF to Office resource files

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

Note:

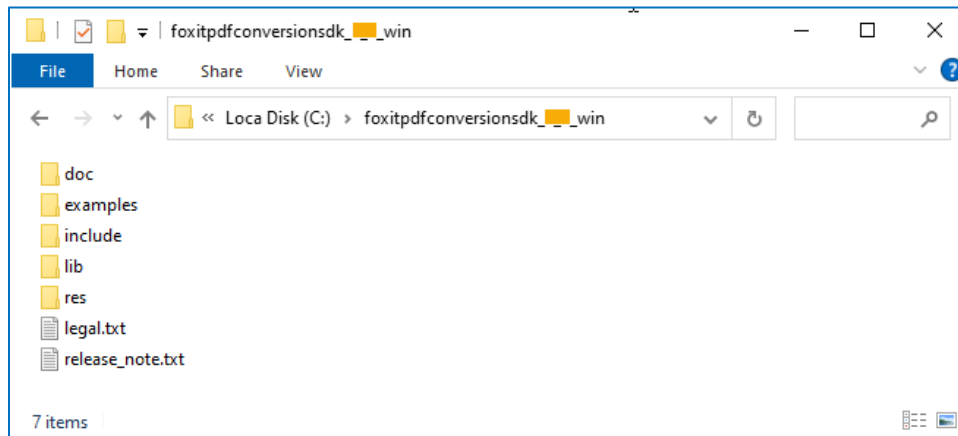
- From version 9.2, it requires Foxit PDF Conversion SDK 1.5 or higher.
- For version 9.0/9.1, it requires Foxit PDF Conversion SDK 1.4 or lower.

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

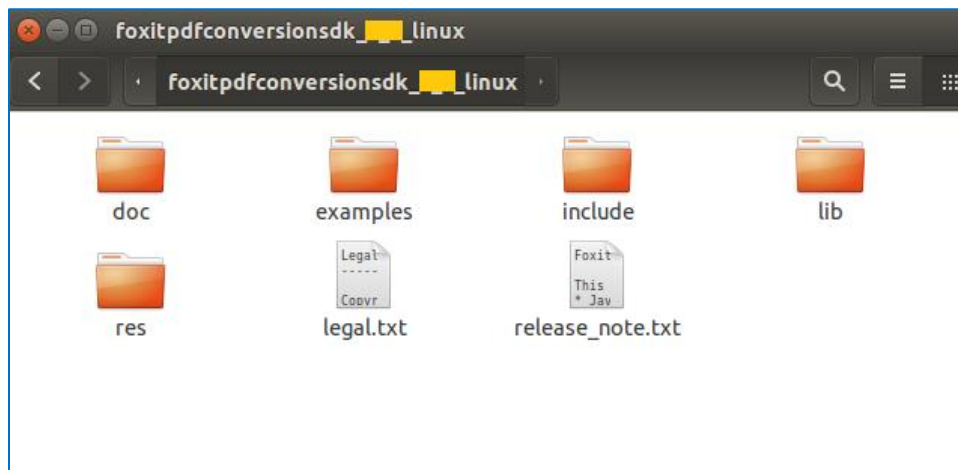
"/foxitpdfconversionsdk_*_Linux/" for Linux x86/x64, and

"/foxitpdfconversionsdk_*_Linux_arm/" for Linux armv7/armv8), and then you can see the resource files for PDF to Office are as follows:

For **Windows**:



For **Linux x86/x64**:



For **Linux armv7/armv8**:

```
localhost:~/Desktop/foxitpdfconversionsdk_*_linux_arm # ls -l --time-style=iso
total 48
drwxr-xr-x 3 root root 4096 05-16 08:12 doc
drwxr-xr-x 3 root root 4096 05-16 08:12 examples
drwxr-xr-x 4 root root 4096 05-16 08:12 include
-rw-rw-rw- 1 root root 21005 05-16 08:12 legal.txt
drwxr-xr-x 2 root root 4096 05-16 08:12 lib
-rw-rw-rw- 1 root root 1834 05-16 08:12 release_note.txt
drwxr-xr-x 3 root root 4096 05-16 08:12 res
```

3.44.3 How to run the pdf2office demo

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

3.44.3.1 Prepare a PDF2Office resource directory

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

3.44.3.2 Configure the demo

For pdf2office demo, you can configure the demo in the "\examples\simple_demo\pdf2office\pdf2office.java" file. Following will configure the demo in "pdf2office.cpp" file on Windows for example. For Linux platform, do the similar configuration with Windows.

Specify the pdf2office engine directory

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

```
// Please ensure the path is valid.  
PDF2Office.initialize("C:/foxitpdfconversionsdk*_win/lib/fpdfconversionsdk_win32.dll");
```

Specify the pdf2office metrics data files

```
PDF2OfficeSettingData setting_data = new PDF2OfficeSettingData();  
// Please ensure the path is valid.  
setting_data.setMetrics_data_folder_path("C:/foxitpdfconversionsdk*_win/res/metrics_data");
```

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

```
setting_data.enable_ml_recognition(false);
```

(Optional) Specify the page range to be converted

```
setting_data.setPage_range(new Range());
```

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

```
setting_data.setInclude_pdf_comments(true);
```

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

```
setting_data.getWord_setting_data().setEnable_retain_page_layout(false);
```

3.44.3.3 Run the demo

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

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

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

3.44.4 How to work with PDF2office API

```
import com.foxit.sdk.PDFException;  
import com.foxit.sdk.common.Library;  
import com.foxit.sdk.common.Progressive;  
import com.foxit.sdk.common.fxcr.FileReaderCallback;  
import com.foxit.sdk.addon.conversion.pdf2office.PDF2Office;  
import com.foxit.sdk.addon.conversion.pdf2office.ConvertCallback;  
import com.foxit.sdk.addon.conversion.pdf2office.PDF2OfficeSettingData;  
import java.io.*;  
  
class CustomConvertCallback extends ConvertCallback {  
    CustomConvertCallback() {}  
  
    @Override  
    public boolean needToPause() {  
        return true;  
    }  
  
    @Override  
    public void progressNotify(int converted_count, int total_count) {  
    }  
}  
  
CustomConvertCallback callback = new CustomConvertCallback();  
Progressive progressive = PDF2Office.startConvertToWord(input_path + "word.pdf", null, output_path +  
"pdf2word_result.docx", setting_data, callback);  
if (progressive.getRateOfProgress() != 100)  
{  
    int state = Progressive.e_ToBeContinued;  
    while (Progressive.e_ToBeContinued == state)  
    {  
        state = progressive.resume();  
    }  
}  
System.out.println("Convert PDF file to Word format file.");
```

```
progressive = PDF2Office.startConvertToExcel(input_path + "excel.pdf", null, output_path +
"pdf2excel_result.xlsx", setting_data, callback);
if (progressive.getRateOfProgress() != 100)
{
    int state = Progressive.e_ToBeContinued;
    while (Progressive.e_ToBeContinued == state)
    {
        state = progressive.resume();
    }
}
System.out.println("Convert PDF file to Excel format file.");

progressive = PDF2Office.startConvertToPowerPoint(input_path + "powerpoint.pdf", null, output_path +
"pdf2powerpoint_result.pptx", setting_data, callback);
if (progressive.getRateOfProgress() != 100)
{
    int state = Progressive.e_ToBeContinued;
    while (Progressive.e_ToBeContinued == state)
    {
        state = progressive.resume();
    }
}
System.out.println("Convert PDF file to PowerPoint format file.");
```

3.45 DWG to PDF Conversion

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

3.45.1 System requirements

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

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

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

SDK Version: Foxit PDF SDK 10.0

3.45.2 DWG To PDF engine files

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

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

3.45.3 How to run the dwg2pdf demo

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

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

Note: For Linux (x86 and x64) and Mac x64, before running the demo, you should configure environment variables.

- For Linux x86 and x64, add the path of the dwg2pdf engine file to `LD_LIBRARY_PATH` environment variable.

```
export LD_LIBRARY_PATH=/dwgtopdf/linux:$LD_LIBRARY_PATH
```

- For Mac x64, add the path of the dwg2pdf engine file to `DWG_ENGINE_PATH` environment variable.

```
export DWG_ENGINE_PATH=/dwgtopdf/mac
```

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

3.45.4 How to convert DWG to PDF

```
import com.foxit.sdk.addon.conversion.DWG2PDFSettingData;
import com.foxit.sdk.addon.conversion.Convert;

// Make sure that SDK has already been initialized successfully.
...
com.foxit.sdk.addon.conversion.DWG2PDFSettingData pdf_setting_data = new com.foxit.sdk.addon.conversion.
DWG2PDFSettingData();
pdf_setting_data.setExport_flags(DWG2PDFSettingData.e_FlagEmbeddedTTF);
pdf_setting_data.setExport_hatches_type(DWG2PDFSettingData.e_DWG2PDFExportHatchesTypeBitmap);
pdf_setting_data.setOther_export_hatches_type(DWG2PDFSettingData.e_DWG2PDFExportHatchesTypeBitmap);
pdf_setting_data.setGradient_export_hatches_type(DWG2PDFSettingData.e_DWG2PDFExportHatchesTypeBitmap);
pdf_setting_data.setSearchable_text_type(DWG2PDFSettingData.e_DWG2PDFSearchableTextTypeNoSearch);
pdf_setting_data.setIs_active_layout(false);
pdf_setting_data.setPaper_width(640); pdf_setting_data.setPaper_height(900);

com.foxit.sdk.addon.conversion.Convert.FromDWG(engine_path, dwg_file_path, output_path, pdf_setting_data);
...
```

3.46 OFD

OFD files, standing for Open Financial Document, are used for storing and exchanging digital financial documents. They are open and XML-based, making them specifically designed for financial documents like contracts, invoices, and statements.

OFD files contain structured data and graphical elements defining the document's layout and content, including text, images, vector graphics, annotations, and other related information. The XML format facilitates easy interpretation, manipulation, and rendering of the document's content.

OFD files provide various benefits, including document integrity, security, and interoperability. They can be digitally signed to ensure authenticity and can be encrypted to protect sensitive information. OFD files also support interactive features like form fields and digital signatures.

To work with OFD files, OFD viewer or editor software that supports the OFD standard is needed. These tools allow you to display, edit, convert, and print the contents of OFD documents.

In essence, OFD files provide a standardized and efficient method for representing financial documents digitally, simplifying the exchange, storage, and management of financial information.

3.46.1 System requirements

Platform: Windows, Linux (x64 and armv8)

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

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

SDK Version: Foxit PDF SDK 10.0

3.46.2 OFD engine file

Please contact Foxit support team or sales team to get the OFD engine file package.

After getting the package, extract it to the desired directory. For example, extract the package to a directory: "**D:/ofd/win**" for Windows, "**ofd/linux64**" for Linux x64, and "**ofd/linuxarm64**" for Linux arm64.

3.46.3 How to run the ofd demo

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

```
// "engine_path" is the path of ofd engine file. Please refer to Developer Guide for more details.  
String engine_path = "D:/ofd/win/x64"; // For Windows x64
```

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

3.46.4 How to implement the conversion between OFD file and PDF file

```
import com.foxit.sdk.common.Library;  
import com.foxit.sdk.addon.conversion.OFDConvertParam;  
import com.foxit.sdk.addon.conversion.Convert;  
...
```



```
// Initialize OFD engine.
Library.initializeOFDEngine(engine_path);

static String src_ofd_path = input_path + "wm_txttiled.ofd";
static String src_pdf_path = input_path + "test.pdf";

// Convert PDF document to OFD document, and convert OFD document to PDF document.
{
    OFDConvertParam convert_param = new OFDConvertParam(false);
    // Convert OFD document to PDF document.
    Convert.fromOFD(src_ofd_path, "", output_path + "ofd2pdf.pdf", convert_param);
    // Convert PDF document to OFD document.
    Convert.toOFD(src_pdf_path, "", output_path + "pdf2ofd.ofd", convert_param);
}
// Release OFD engine.
Library.releaseOFDEngine();
```

3.46.5 How to render OFD page

```
import com.foxit.sdk.common.Image;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.Bitmap;
import com.foxit.sdk.addon.ofd.*;
import com.foxit.sdk.common.Progressive;
import com.foxit.sdk.common.fxcrd.Matrix2D;
import com.foxit.sdk.common.fxcrd.RectI;
...

// Initialize OFD engine.
Library.initializeOFDEngine(engine_path);

// Render OFD document to bitmap.
{
    String render_file_path = input_path + "content_flag.ofd";
    OFDDoc doc = new OFDDoc(render_file_path, "");
    OFDPage ofd_page = doc.getPage(0);
    // Get the size of the page.
    float w = ofd_page.getWidth();
    float h = ofd_page.getHeight();
    Bitmap bitmap = new Bitmap((int)w, (int)h, Bitmap.e_DIBArgb, null, 0);
    RectI fRect = new RectI(0, (int)h, (int)w, 0);
    bitmap.fillRect(0xFFFFFFFF, fRect);
    // Get the display matrix of the page.
    Matrix2D matrix_1 = ofd_page.getDisplayMatrix(0, 0, (int)w, (int)h, e_Rotation0);
    OFDRenderer ofd_render = new OFDRenderer(bitmap);

    Progressive progressive = ofd_render.startRender(ofd_page, matrix_1);
    String sSaveFilePath = output_path + "renderBitmap.bmp";
    // Add the bitmap to image and save the image.
    Image image = new Image();
    image.addFrame(bitmap);
    image.saveAs(sSaveFilePath);
}
```

```
ofd_page.release();  
doc.release();  
}  
  
// Release OFD engine.  
Library.releaseOFDEngine();
```

3.47 Paragraph Editing

Foxit PDF SDK offers a versatile set of tools for developers to fine-tune and customize text in PDF documents. The paragraph editing module provides complex adjustments, joining, and splitting functions that allow users to have precise control over the content of the document. The features are complemented by an intuitive UI implementation that facilitates efficient editing, ensuring a seamless and customized experience for managing text paragraphs.

The paragraph editing functionality revolves around two core modules, the **ParagraphEditing** module, and the **JoinSplit** module.

The **ParagraphEditing** module is designed to offer a variety of text editing operations, enabling users to easily perform the following actions according to their specific requirements:

- **Insert Text:** Insert new content at specific locations, allowing for customization of the document's precise layout.
- **Delete Text:** Delicately remove paragraphs or characters, enabling highly customized content trimming.
- **Modify Text:** Adjust existing text, including its content and formatting, to suit different editing styles.
- **Format Adjustment:** Support fine adjustments to paragraph formats and text styles, allowing for more accurate typesetting.

The **JoinSplit** module contains four vital operation types to support more complex text processing requirements:

- **Join:** Integrate multiple text blocks, enhancing content layout and overall document consistency.
- **Split:** Finely split text blocks, providing flexibility to manage various sections of the document.
- **Link:** Establish connections between text blocks, ensuring consistency in associated content.
- **Unlink:** Disconnect links between text blocks, offering more control over editing.

3.47.1 System requirements

Platform: Windows, Linux, Mac

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

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

SDK Version: Foxit PDF SDK 10.0

3.47.2 How to work with paragraph editing

```
import com.foxit.sdk.addon.pageeditor.*;
import com.foxit.sdk.common.Library;
import com.foxit.sdk.common.fxcr.*;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.common.fxcr.RectF;
import com.foxit.sdk.common.fxcr.RectFArray;
import com.foxit.sdk.common.fxcr.Matrix2D;
import java.io.File;

...

class IParagraphEditingProviderCallback extends ParagraphEditingProviderCallback
{
    private int current_page_index = 0;

    @Override
    public void release() {

    }

    @Override
    public Matrix2D getRenderMatrix(PDFDoc document,int page_index) {

        try {
            PDFPage currentpage = document.getPage(page_index);
            int width = (int) currentpage.getWidth();
            int height = (int) currentpage.getHeight();
            Matrix2D matrix = new Matrix2D();
            matrix = currentpage.getDisplayMatrix(0, 0, width, height, currentpage.getRotation());
            return matrix;
        } catch (PDFException e) {

            e.printStackTrace();
            return null;
        }
    }

    @Override
    public Object getPageViewHandle(PDFDoc document,int page_index) {
```

```
    return null;
}

@Override
public RectF getClientRect(PDFDoc document) {
    return new RectF();
}

@Override
public float getScale(PDFDoc document,int page_index) {
    return 1;
}

@Override
public boolean gotoPageView(PDFDoc document,int page_index, float left, float top) {
    return true;
}

@Override
public Int32Array getVisiblePageIndexArray(PDFDoc document) {
    Int32Array page_array = new Int32Array();
    page_array.add(0);
    return page_array;
}

@Override
public RectF getPageVisibleRect(PDFDoc document,int page_index) {
    return new RectF();
}

@Override
public RectF getPageRect(PDFDoc document,int page_index) {
    try {
        PDFPage currentpage = document.getPage(page_index);
        int width = (int) currentpage.getWidth();
        int height = (int) currentpage.getHeight();
        RectF rect = new RectF(0,height,width,0);
        return rect;
    } catch (PDFException e) {

        e.printStackTrace();
        return null;
    }
}

@Override
public int getCurrentPageIndex(PDFDoc document) {
    return current_page_index;
}

@Override
```

```
public int getRotation(PDFDoc document,int page_index) {
    return 0;
}

@Override
public void invalidateRect(PDFDoc doc,int page_index, RectFArray rectFArray) {
}

@Override
public void addUndoItem(ParagraphEditingUndoItem paragraphEditingUndoItem) {
}

@Override
public void setDocChangeMark(PDFDoc pdfDoc) {
}

@Override
public void notifyTextInputReachLimit(PDFDoc doc,int page_index) {
}

public IParagraphEditingProviderCallback(int page_index)
{
    current_page_index = page_index;
}
}

...

page.startParse(0, null, false);
float height = page.getHeight();
if (error_code != e_ErrSuccess)
{
    System.out.println(String.format("The Doc [%s] Error: %d\n", input_file, error_code));
    return;
}
int page_index = page.getIndex();
IParagraphEditingProviderCallback callback = new IParagraphEditingProviderCallback(page_index);
ParagraphEditingMgr editingMgr = new ParagraphEditingMgr(callback, doc);
ParagraphEditing editing = editingMgr.getParagraphEditing();
if (editing.isEmpty() == true)
    return;
editing.activate();

editing.startEditing(0, new PointF(95, height - 728), new PointF(95, height - 728));
editing.setFontSize(24);
editing.setUnderline(true);
editing.insertText("InsertText_Paragraph_editing");
editing.deactivate();
doc.saveAs(output_file, PDFDoc.e_SaveFlagNoOriginal);

// Join&split
JoinSplit joinSplit = editingMgr.getJoinSplit();
```

```
joinSplit.activate();
joinSplit.onLButtonDown(0, new PointF(289, 659));
joinSplit.onLButtonUp(0, new PointF(289, 659));
joinSplit.splitBoxes();
joinSplit.deactivate();
String output_path_split = output_path + "Split_Boxes.pdf";
doc.saveAs(output_path_split, PDFDoc.e_SaveFlagNoOriginal);

joinSplit.activate();
joinSplit.onLButtonDown(0, new PointF(307, height - 637));
joinSplit.onLButtonUp(0, new PointF(307, height - 637));
joinSplit.onLButtonDown(0, new PointF(307, height - 453));
joinSplit.onLButtonUp(0, new PointF(307, height - 453));
joinSplit.joinBoxes();
joinSplit.deactivate();
String output_path_join = output_path + "Join_Boxes.pdf";
doc.saveAs(output_path_join, PDFDoc.e_SaveFlagNoOriginal);
```

3.48 3D Rendering

3D Rendering in PDFs helps convert 3D models into 2D images or animations, which can be embedded in the document. The 3D models can be created using specialized 3D modeling software or CAD (Computer-Aided Design) software and can then be rendered into a 2D format that can be easily viewed within a PDF.

3D Annotation is a feature that allows users to add contextual information to specific parts of these 3D models within the PDF. The annotations can include text, images, and even links to external resources, providing more detailed information and insights about the model.

3.48.1 System requirements

Platform: Windows

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

License Key requirement: '3D' module permission in the license key

SDK Version: Foxit PDF SDK 10.0

Note: If you are using a Windows 7 system, you may need to visit this link <https://support.microsoft.com/en-us/help/4019990/update-for-the-d3dcompiler-47-dll-component-on-windows> to download and install the "D3DCOMPILER_47.dll". If you do not do this, you may encounter errors when using the 3D module.

3.48.2 How to display the 3D annotation

```
import com.foxit.sdk.addon.pdf3d.*;
```

```
// Load PDF document.
```

```
...
PDF3DContext pdf_context = new PDF3DContext(pdf_doc);

// Get the 3d annotation instance array.
PDF3DAnnotInstanceArray annot_data_arr = pdf_context.getPage3DAnnotArray(0);
if(annot_data_arr.getSize() == 0 ) return;
// Class parameter.
PDF3DAnnotInstance annotData = annot_data_arr.getAt(0);
// Activate the canvas to display the 3d annotation. Pass in a "window handle" to embed canvas.
annotData.activateCanvas(window handle);
```

3.48.3 How to set render mode and controller

```
import com.foxit.sdk.addon.pdf3d.*;

// Rotate to view 3D annotations.
annotData.setController(e_ControllerRotate);

// Render 3D annotations as transparent.
annotData.setRenderMode(e_RenderModeTransparent);
```

FAQ

1. 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 **PDFPage.getGraphicsObjectAtPoint** 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:

```
import com.foxit.sdk.PDFException;
import com.foxit.sdk.common.fxcrf.PointF;
import com.foxit.sdk.pdf.PDFDoc;
import com.foxit.sdk.pdf.PDFPage;
import com.foxit.sdk.pdf.graphics.*;

import static com.foxit.sdk.common.Constants.*;
import static com.foxit.sdk.pdf.PDFDoc.*;
import static com.foxit.sdk.pdf.graphics.GraphicsObject.*;
...

public class graphics_objects {
    ...
    static bool ChangeTextObjectContent() throws PDFException {

        String input_file = input_path + "AboutFoxit.pdf";
        PDFDoc doc = new PDFDoc(input_file);
        int error_code = doc.load(null);
        if (error_code != e_ErrSuccess) {
            System.out.println(String.format("The Doc [%s] Error: %d\n", input_file, error_code);
            return false;
        }
        // Get original shading objects from the first PDF page.
        PDFPage original_page = doc.getPage(0);
        original_page.startParse((e_ParsePageNormal, null, false);
        PointF pointf = new PointF(92, 762);
        GraphicsObjectArray arr = original_page.getGraphicsObjectsAtPoint(pointf, 10, e_TypeText);
        for(int i = 0; i<arr.getSize(); i++) {
            GraphicsObject graphobj = arr.getAt(i);
            TextObject textobj = graphobj.getTextObject();
            textobj.setText("Foxit Test");
        }
    }
}
```



```
}  
original_page.generateContent();  
  
String output_directory = output_path + "graphics_objects/";  
String output_file = output_directory + "After_revise.pdf";  
doc.saveAs(output_file, e_SaveFlagNormal);  
  
return true;  
}  
...  
}
```

2. 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 "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 "Image.addFrame", and it does not support the image formats of JPX, GIF and TIF.

3. Why do I encounter "Fail to initialize the engine file or cannot load the engine file" for OCR and DWG2PDF modules on Windows 7 when running the corresponding simple demos, even if the engine files have been upgraded to the latest and the simple demos have configured the engine path correctly?

For Windows 7, you need to copy the dll files starting with **api-ms-win*** and the **ucrtbase.dll** in the engine directory to the system directory.

- If you are using a 32-bit engine and running on a 32-bit system, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/System32".
- If you are using a 32-bit engine and running on a 64-bit system, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/SysWOW64".
- If you are using a 64-bit engine, you need to copy the api-ms-win*.dll files and ucrtbase.dll from the engine directory to "C:/Windows/System32".

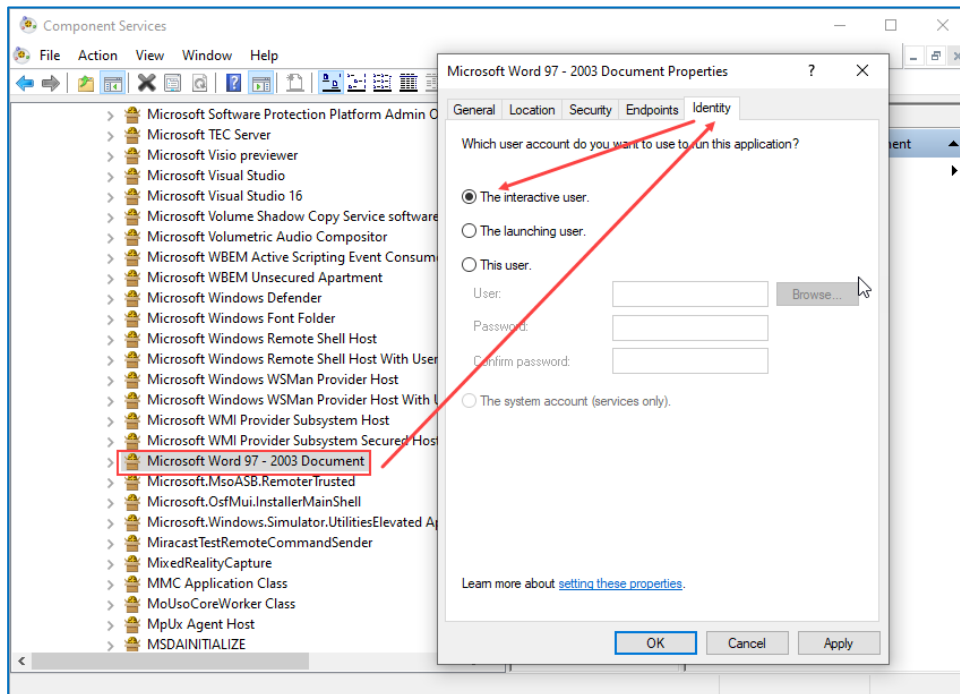
4. How to run the Office2PDF functionality in Windows services?

To run the Office2PDF functionality in Windows services, you need to configure the Office Component Services and permissions.

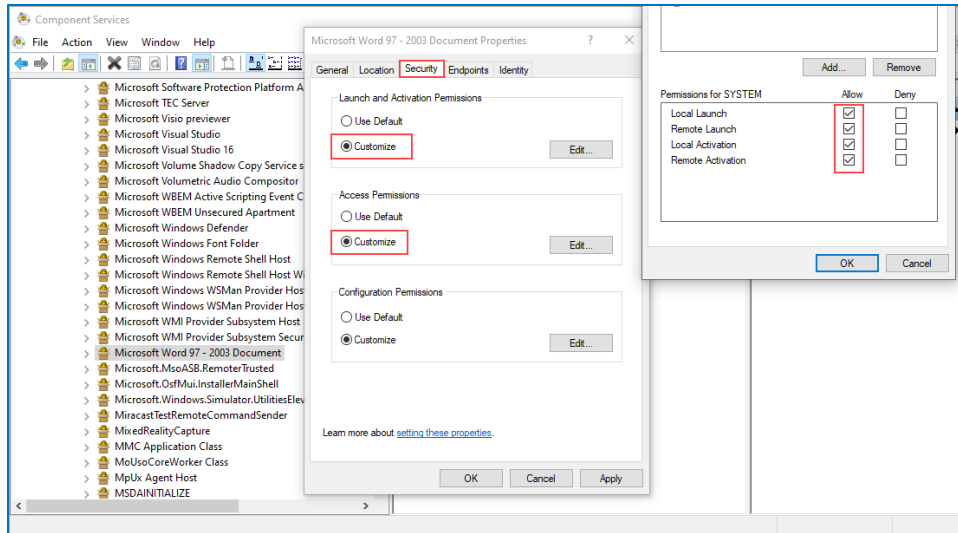
Take the Word component as an example.

- 1) Press **Win+R**, and then type **Dcomcnfg** to open Component Services, find [Component Services] -> [Computers] -> [My Computer] -> [DCOM Config] -> [Microsoft Word 97-2003 Document], right-click it and select Properties. Choose [Identity], and set it to "**The interactive user**".

Note: if you can't find [Microsoft Word 97-2003 Document] with **Dcomcnfg** command, you can try to use the "**comexp.msc -32**" command.



- 2) Set permissions. click [Security], set the [Launch and Activation Permissions] and [Access Permissions] to **Customize**. Click **Edit**, and add the current login account of the system and enable all permissions.



- 3) After finishing the above settings, the Word2PDF functionality can be run in the Windows services.

APPENDIX

Supported JavaScript List

Objects' property or method

Object	Properties/Method Names	Minimum Supported SDK Version
annotation properties	alignment	V7.0
	author	V7.0
	contents	V7.0
	creationDate	V7.0
	fillColor	V7.0
	hidden	V7.0
	modDate	V7.0
	name	V7.0
	opacity	V7.0
	page	V7.0
	readOnly	V7.0
	rect	V7.0
	richContents	V7.1
	rotate	V7.0
	strokeColor	V7.0
	textSize	V7.0
	type	V7.0
	AP	V9.0
	arrowBegin	V9.0
	arrowEnd	V9.0
	attachIcon	V9.0
	attachment	V9.0
	borderEffectIntensity	V9.0
	borderEffectStyle	V9.0
	callout	V9.0
	caretSymbol	V9.0
	dash	V9.0
	delay	V9.0
	doc	V9.0
	doCaption	V9.0
	gestures	V9.0
	inReplyTo	V9.0
	intent	V9.0
	leaderExtend	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	leaderLength	V9.0
	lineEnding	V9.0
	lock	V9.0
	notelcon	V9.0
	noView	V9.0
	point	V9.0
	points	V9.0
	popupOpen	V9.0
	popupRect	V9.0
	print	V9.0
	quads	V9.0
	refType	V9.0
	richDefaults	V9.0
	seqNum	V9.0
	soundIcon	V9.0
	style	V9.0
	subject	V9.0
	textFont	V9.0
	toggleNoView	V9.0
	vertices	V9.0
	width	V9.0
annotation method	destroy	V7.0
	getProps	V9.0
	setProps	V9.0
	getStateInModel	V9.0
app properties	activeDocs	V4.0
	calculate	V4.0
	formsVersion	V4.0
	fs	V4.0
	fullscreen	V4.0
	language	V4.2
	platform	V4.0
	runtimeHighlight	V4.0
	viewerType	V4.0
	viewerVariation	V4.0
	viewerVersion	V4.0
	printerNames	V8.4
	runtimeHighlightColor	V8.4
	constants	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
app methods	alert	V4.0
	beep	V4.0
	browseForDoc	V4.0
	clearInterval	V4.0
	clearTimeout	V4.0
	launchURL	V4.0
	mailMsg	V4.0
	response	V4.0
	setInterval	V4.0
	setTimeout	V4.0
	popupMenu	V4.0
	execDialog	V8.4
	execMenuItem	V8.4
	newDoc	V8.4
	openDoc	V8.4
	popupMenuEx	V8.4
	addMenuItem	V8.4
	addSubMenu	V8.4
	addToolButton	V8.4
	removeToolButton	V8.4
	listMenuItems	V8.4
	trustedFunction	V8.4
	beginPriv	V8.4
	endPriv	V8.4
color properties	black	V4.0
	blue	V4.0
	cyan	V4.0
	dkGray	V4.0
	gray	V4.0
	green	V4.0
	ltGray	V4.0
	magenta	V4.0
	red	V4.0
	transparent	V4.0
	white	V4.0
	yellow	V4.0
color methods	convert	V4.0
	equal	V4.0
document properties	author	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	baseURL	V4.0
	bookmarkRoot	V7.0
	calculate	V4.0
	Collab	V4.0
	creationDate	V4.0
	creator	V4.0
	delay	V4.0
	dirty	V4.0
	documentFileName	V4.0
	external	V4.0
	filesize	V4.0
	icons	V4.0
	info	V4.0
	keywords	V4.0
	modDate	V4.0
	numFields	V4.0
	numPages	V4.0
	pageNum	V4.0
	path	V4.0
	producer	V4.0
	subject	V4.0
	title	V4.0
	URL	V8.4
	dataObjects	V8.4
	hostContainer	V8.4
	templates	V8.4
	media	V8.4
	dynamicXFAForm	V8.4
	mouseX	V8.4
	mouseY	V8.4
	pageWindowRect	V8.4
	securityHandler	V8.4
	zoom	V8.4
	zoomType	V8.4
	layout	V8.4
	xfa	V8.4
document methods	addAnnot	V7.0
	addField	V4.0
	addIcon	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	calculateNow	V4.0
	deletePages	V4.0
	exportAsFDF	V4.0
	flattenPages	V7.1
	getAnnot	V7.0
	getAnnots	V7.0
	getField	V4.0
	getIcon	V4.0
	getNthFieldName	V4.0
	getOCGs	V4.0
	getPageBox	V4.0
	getPageNthWord	V4.0
	getPageNthWordQuads	V4.0
	getPageNumWords	V4.0
	getPageRotation	V7.0
	getPrintParams	V4.0
	getURL	V4.0
	importAnFDF	V4.0
	insertPages	V6.2
	mailForm	V4.0
	print	V4.0
	removeField	V4.0
	replacePages	V6.2
	resetForm	V4.0
	submitForm	V4.0
	mailDoc	V4.0
	addWatermarkFromFile	V8.4
	addWatermarkFromText	V8.4
	getPageLabel	V8.4
	setPageLabels	V8.4
	gotoNamedDest	V8.4
	saveAs	V8.4
	scroll	V8.4
	setPageTabOrder	V8.4
	selectPageNthWord	V8.4
	syncAnnotScan	V8.4
	getAnnot3D	V8.4
	getAnnots3D	V8.4
	addLink	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	removeLinks	V8.4
	getLinks	V8.4
	importIcon	V8.4
	removeIcon	V8.4
	addWeblinks	V8.4
	removeWeblinks	V8.4
	closeDoc	V8.4
	exportDataObject	V8.4
	importDataObject	V8.4
	removeDataObject	V8.4
	getDataObject	V8.4
	embedDocAsDataObject	V8.4
	createTemplate	V8.4
	removeTemplate	V8.4
	getTemplate	V8.4
	exportAsText	V8.4
	importTextData	V8.4
	exportAsXFDF	V8.4
	importAnXFDF	V8.4
	exportAsXFDFStr	V8.4
	extractPages	V8.4
	movePage	V8.4
	newPage	V8.4
	getOCGOrder	V8.4
	setOCGOrder	V8.4
	setPageBoxes	V8.4
	setPageRotations	V8.4
	setPageTransitions	V9.1
	getPageTransition	V9.1
event properties	change	V4.0
	changeEx	V4.0
	commitKey	V4.0
	fieldFull	V4.0
	keyDown	V4.0
	modifier	V4.0
	name	V4.0
	rc	V4.0
	selEnd	V4.0
	selStart	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	shift	V4.0
	source	V4.0
	target	V4.0
	targetName	V4.0
	type	V4.0
	value	V4.0
	willCommit	V4.0
event methods	add	V9.0
field properties	alignment	V4.0
	borderStyle	V4.0
	buttonAlignX	V4.0
	buttonAlignY	V4.0
	buttonFitBounds	V4.0
	buttonPosition	V4.0
	buttonScaleHow	V4.0
	buttonScaleWhen	V4.0
	calcOrderIndex	V4.0
	charLimit	V4.0
	comb	V4.0
	commitOnSelChange	V4.0
	currentValueIndices	V4.0
	defaultValue	V4.0
	doNotScroll	V4.0
	doNotSpellCheck	V4.0
	delay	V4.0
	display	V4.0
	doc	V4.0
	editable	V4.0
	exportValues	V4.0
	hidden	V4.0
	fileSelect	V4.0
	fillColor	V4.0
	lineWidth	V4.0
	highlight	V4.0
	multiline	V4.0
	multipleSelection	V4.0
	name	V4.0
	numItems	V4.0
	page	V4.0

Object	Properties/Method Names	Minimum Supported SDK Version
	password	V4.0
	print	V4.0
	radiosInUnison	V4.0
	readonly	V4.0
	rect	V4.0
	required	V4.0
	richText	V4.0
	rotation	V4.0
	strokeColor	V4.0
	style	V4.0
	textColor	V4.0
	textFont	V4.0
	textSize	V4.0
	type	V4.0
	userName	V4.0
	value	V4.0
	valueAsString	V4.0
	richValue	V9.0
	submitName	V9.0
field methods	browseForFileToSubmit	V4.0
	buttonGetCaption	V4.0
	buttonGetIcon	V4.0
	buttonSetCaption	V4.0
	buttonSetIcon	V4.0
	checkThisBox	V4.0
	clearItems	V4.0
	defaultIsChecked	V4.0
	deleteItemAt	V4.0
	getArray	V4.0
	getItemAt	V4.0
	insertItemAt	V4.0
	isBoxChecked	V4.0
	isDefaultChecked	V4.0
	setAction	V4.0
	setFocus	V4.0
	setItems	V4.0
	buttonImportIcon	V9.0
	getLock	V9.0
	setLock	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	signatureGetModifications	V9.0
	signatureGetSeedValue	V9.0
	signatureInfo	V9.0
	signatureSetSeedValue	V9.0
	signatureSign	V9.0
	signatureValidate	V9.0
global methods	setPersistent	V4.0
Icon properties	name	V4.0
util methods	printd	V4.0
	printf	V4.0
	printx	V4.0
	scand	V4.0
	iconStreamFromIcon	V9.0
identity properties	loginName	V4.2
	name	V4.2
	corporation	V4.2
	email	V4.2
collab properties	user	V6.2
ocg properties	name	V6.2
ocg methods	setAction	V6.2
bookmark properties	color	V8.4
	open	V8.4
	name	V8.4
	parent	V8.4
	children	V8.4
	language	V8.4
	style	V8.4
	platform	V8.4
bookmark methods	createChild	V8.4
	insertChild	V8.4
	execute	V8.4
	setAction	V8.4
	remove	V8.4
certificate properties	binary	V8.4
	issuerDN	V8.4
	keyUsage	V8.4
	MD5Hash	V8.4
	privateKeyValidityEnd	V8.4
	privateKeyValidityStart	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	SHA1Hash	V8.4
	serialNumber	V8.4
	subjectCN	V8.4
	subjectDN	V8.4
	validityEnd	V8.4
	validityStart	V8.4
RDN properties	c	V8.4
	cn	V8.4
	e	V8.4
	l	V8.4
	o	V8.4
	ou	V8.4
	st	V8.4
security properties	handlers	V9.0
security methods	getHandler	V9.0
	importFromFile	V9.0
securityHandler properties	appearances	V9.0
	isLoggedIn	V9.0
	loginName	V9.0
	loginPath	V9.0
	name	V9.0
	uiName	V9.0
securityHandler methods	login	V9.0
	logout	V9.0
	newUser	V9.0
signatureInfo properties	objValidity	V9.0
	idValidity	V9.0
	idPrivValidity	V9.0
	docValidity	V9.0
	byteRange	V9.0
	verifyHandlerUIName	V9.0
	verifyHandlerName	V9.0
	verifyDate	V9.0
	subFilter	V9.0
	statusText	V9.0
	status	V9.0
	reason	V9.0
	name	V9.0
	mdp	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	location	V9.0
	handlerUIName	V9.0
	handlerUserName	V9.0
	handlerName	V9.0
	dateTrusted	V9.0
	date	V9.0
search properties	attachments	V9.0
	bookmarks	V9.0
	docText	V9.0
	ignoreAccents	V9.0
	markup	V9.0
	matchCase	V9.0
	matchWholeWord	V9.0
	maxDocs	V9.0
	proximity	V9.0
	stem	V9.0
	wordMatching	V9.0
	ignoreAsianCharacterWidth	V9.0
search methods	query	V9.0
	addIndex	V9.0
	removeIndex	V9.0
link properties	borderColor	V8.4
	borderWidth	V8.4
	highlightMode	V8.4
	rect	V8.4
link methods	setAction	V8.4
app.media properties	align	V8.4
	canResize	V8.4
	ifOffScreen	V8.4
	over	V8.4
	windowType	V8.4
app.media methods	createPlayer	V8.4
	openPlayer	V8.4
doc.media methods	getOpenPlayers	V8.4
Playerargs properties	doc	V8.4
	annot	V8.4
	rendition	V8.4
	URL	V8.4
	mimeType	V8.4

Object	Properties/Method Names	Minimum Supported SDK Version
	settings	V8.4
	events	V8.4
MediaPlayer properties	isOpen	V8.4
	isPlaying	V8.4
	settings	V8.4
	visible	V8.4
MediaPlayer methods	close	V8.4
	play	V8.4
	seek	V8.4
	stop	V8.4
MediaSettings properties	autoPlay	V8.4
	baseURL	V8.4
	bgColor	V8.4
	bgOpacity	V8.4
	duration	V8.4
	floating	V8.4
	page	V8.4
	repeat	V8.4
	showUI	V8.4
	visible	V8.4
	volume	V8.4
	windowType	V8.4
floating properties	align	V8.4
	over	V8.4
	canResize	V8.4
	hasClose	V8.4
	hasTitle	V8.4
	title	V8.4
	ifOffScreen	V8.4
	rect	V8.4
eventListener methods	afterClose	V9.0
	afterPlay	V9.0
	afterReady	V9.0
	afterSeek	V9.0
	afterStop	V9.0
	onClose	V9.0
	onPlay	V9.0
	onReady	V9.0
	onSeek	V9.0

Object	Properties/Method Names	Minimum Supported SDK Version
	onStop	V9.0
Template properties	hidden	V9.1
	name	V9.1
Template method	spawn	V9.1
span properties	alignment	V9.1
	fontFamily	V9.1
	fontStretch	V9.1
	fontWeight	V9.1
	fontStyle	V9.1
	strikethrough	V9.1
	subscript	V9.1
	superscript	V9.1
	text	V9.1
	textColor	V9.1
	textSize	V9.1
	underline	V9.1
soap properties	wireDump	V9.1
Soap method	request	V9.1
	streamDigest	V9.1
	streamEncode	V9.1
	streamFromString	V9.1
	stringFromStream	V9.1
hostContainer method	postMessage	V9.2
Fullscreen properties	transitions	V9.2
	defaultTransition	V9.2
	loop	V9.2
	timeDelay	V9.2
	useTimer	V9.2
	isFullScreen	V9.2

Global methods

Method Names	Minimum Supported SDK Version
AFNumber_Format	V4.0
AFNumber_Keystroke	V4.0
AFPercent_Format	V4.0
AFPercent_Keystroke	V4.0
AFDate_FormatEx	V4.0
AFDate_KeystrokeEx	V4.0
AFDate_Format	V4.0

Method Names	Minimum Supported SDK Version
AFDate_Keystroke	V4.0
AFTime_FormatEx	V4.0
AFTime_KeystrokeEx	V4.0
AFTime_Format	V4.0
AFTime_Keystroke	V4.0
AFSpecial_Format	V4.0
AFSpecial_Keystroke	V4.0
AFSpecial_KeystrokeEx	V4.0
AFSimple	V4.0
AFMakeNumber	V4.0
AFSimple_Calculate	V4.0
AFRange_Validate	V4.0
AFMergeChange	V4.0
AFParseDateEx	V4.0
AFExtractNums	V4.0

REFERENCES

[1] PDF reference 1.7

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51502

[2] PDF reference 2.0

<https://www.iso.org/standard/63534.html>

[3] Foxit PDF SDK API reference

sdk_folder/doc/Foxit PDF SDK Java API Reference.html

Note: sdk_folder is the directory of unzipped package.

SUPPORT

Foxit Support

In order to provide you with a more personalized support for a resolution, please log in to your [Foxit account](#) and submit a ticket so that we can collect details about your issue. We will work to get your problem solved as quickly as we can once your ticket is routed to our support team.

You can also check out our [Support Center](#), choose Foxit PDF SDK which also has a lot of helpful articles that may help with solving your issue.

Phone Support:

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