Using Foxit on the Android Platform

Foxit® Embedded PDF SDK

Prerequisites

Developer Audience

This document is targeted towards Android developers using the SDK to add PDF functionality to Android applications. It assumes that the developer is familiar with C/C++ and Java.

Supported Environments

Platform	Operating System	Compiler
Android	Android 2.2 or newer.	android-ndk-r7 or newer.

SDK vs. NDK

The Android platform provides two distinct development environments, the Android SDK and the Android NDK.

The SDK operates at the Java layer and compiles Java/XML sources to byte code used by the Dalvik Virtual Machine.

The Android NDK is a cross compilation environment for C/C++ sources, which was designed to take advantage of the Java Native Interface and expose native functions to the Java.

The Foxit Embedded SDK is a native library, so Android developers can only interface with this native library by using the NDK to export library functions for using in Java and the Android SDK. Since revision 5 of the Android NDK, it is also possible to write complete C/C++ applications for the Android platform without interfacing with the Java layer.

Android Demos

These basic demos use Foxit Embedded SDK technology to securely display, search, fill in forms, and annotate Portable Document Format (PDF). It is NOT a fully-featured application that shows all the capabilities of the Foxit Embedded SDK. The code is designed to give the audience a sample of one possible framework (out of many) for using the Foxit Embedded SDK on Android. The aim is to allow the audience, upon understanding the framework, to add other functionalities easily. Demos and sample code are provided to help developers get up to speed quickly.
For help with using Integrated Development Environments (IDEs) other than Eclipse for compiling this demo, please contact support@foxitsoftware.com

Setting up and Running the Demo Project

1) Download and install the Eclipse IDE (http://www.eclipse.org/), the Android SDK, ADT plugin for Eclipse, and the Android NDK (http://developer.android.com/sdk/index.html).
   a) For Windows use, also download and install Cygwin (http://www.cygwin.com/). During Cygwin setup, make sure to include the "Devel -> make" package.

2) Download the Foxit embedded SDK Package.
3) Extract the Foxit embedded SDK Package to any directory.
4) Place the Foxit embedded SDK library and header files in fpdfemb_android/examples/demos/bin and include directory.
5) Build the NDK layer.
   a) Open the Android.mk makefile in fpdfemb_android/examples/demos/demo(like "demo_view")/jni/ in a text editor and fill in the Foxit library name in the area designated for LOCAL_LDLIBS, dropping the lib prefix:

      The demo is shipped as:
      LOCAL_LDLIBS +=../bin/# fill in library name here

      To add downloaded libfoxit.a from step 2, fill in as:
      LOCAL_LDLIBS :=../bin/libfoxit.a

   If the library provide is not named "libfoxit.a" please adjust accordingly.

b) Open Cygwin (Windows), or a terminal (Linux based), and navigate to the fpdfemb_android/examples/demos/demo(like "demo_view") directory. Run "ndk-build –B" to build the NDK/JNI layer.

Example:

me@myStation /myProjectPath/ > ndk-build –B

This assumes that the ndk directory is part of the $PATH environment variable. The command can also be qualified with the path to the NDK directory.

c) The "ndk-build" script will automatically place the finished NDK layer in the form of a shared object (.so) in the fpdfemb_android/examples/demos/demo(like "demo_view")/libs/armeabi/ directory.

6) Import the project into Eclipse through File->Import->Existing Project into Workspace, and choose the directory where the demo was extracted.
7) Eclipse builds automatically.
   a) If the NDK/JNI code is changed, it will need to be rebuilt by following steps 5b and 5c. After rebuilding the Eclipse project must be cleaned (Project -> Clean) to allow Eclipse to rebuild your sample. Hairy and unwanted things can occur if this is not done.

NOTE: If you encounter this error message in the "Console" tab in Eclipse,

ERROR: Unable to open class file [full path to extracted demo files]\gen\com\[foxit demo]\frontend\R.java: No such file or directory.

Try regenerating the entire \gen folder by making a change to one of the files. For example,

a) In Eclipse, click on /res/layout/main.xml
b) Make the following change and save it,
   Android:layout_height="fill_parent" to
   Android:layout_height="fill"
c) Now change it back to "fill_parent" and save it. This results in no change to main.xml but you should have generated a new /gen folder.
d) The demo project should build now.
e) If the project does build try Step 7a to clean the project resources.

8) Copy the test files to the android device or emulator. Below is a list of files that are needed to run the demos. Make sure all of the files are present on the SD Card ("\mnt\sdcard") or the demo may not run properly. Most files can be found in "fpdfemb_android/examples/test_files" or "fpdfemb_android/examples" directory.

   Test documents: FoxitBigPreview.pdf, FoxitForm.pdf, FoxitText.pdf, and readme.pdf
   Annotation Files: FoxitLogo.jpg, FoxitForm.pdf
   External font: DroidSansFallback.ttf

Below is a list of generated files. If a demo generates, it will be stored in its own folder like "/data/data/com.foxitsample.annotations/FoxitSaveAnnotation.pdf".

Generated Files: FoxitSaveAnnotation.pdf, FormXml.xml, FoxitEncryptd.pdf, FoxitDecryptd.pdf, FoxitPKIEncryptd.pdf, and FoxitPKIDecryptd.pdf

Attentions:

All demos require the .lib File should be placed in the demos/bin directory and the .h files to be in the include directory.

9) To make the modules work in the demo, please update the parameters of the function named EMBJavaSupport.FSUnlock in the WrapPDFFunc.java with the corresponding License ID and Unlock Code specified in the sn.txt in the fpdfemb_android_lib.zip package.
10) Push the finished foxitSample apk to a device/emulator.
    a) Make sure you have a device/emulator ready either by firing off an Android Virtual Device or having an Android phone/tablet plugged in with Settings->Applications->Development->USB Debugging enabled.
    b) In Eclipse, choose Run->Run to push the foxitSample onto the device. The sample will automatically launch.

Note: If you encounter this error message in the "Console" tab in Eclipse,

"Android requires .class compatibility set to 5.0. Please fix project properties"

Try fixing the project properties. Right click on the demo project, select Android Tools, and then select Fix Properties.

Implementation/Design Notes

The Foxit Embedded SDK is:

a) A C interface to the Foxit PDF technology library that heavily relies on pointers.
b) Open ended with regards to memory. Users are required to choose from several memory management schemes on initialization.
c) Open ended with regards to error handling. There are multiple ways to handle error codes returned by Foxit Embedded SDK functions.

Foxit Embedded SDK users have to decide how and where to handle each of these aspects.

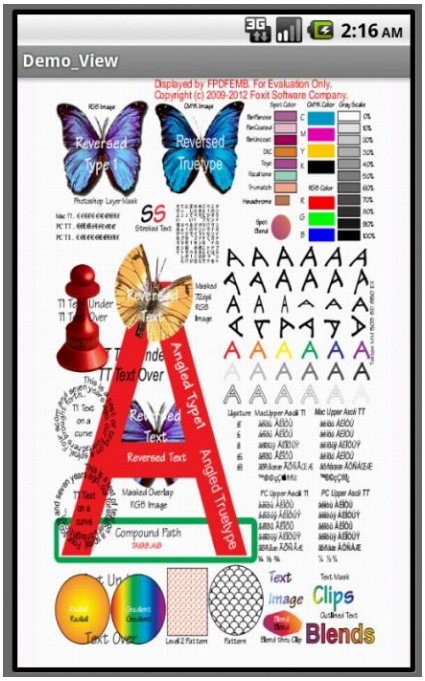
In the Android demo,

a) The C interface is implemented in the NDK/JNI layer, where pointers are returned directly as integer handles to the Java layer, and the Java layer is expected to know about and handle each pointer passed back appropriately.
b) The demo uses a simple memory management scheme: give a finite amount of memory for the PDF engine to use. The memory initialization requirement is handled in the NDK/JNI layer using memory from the system heap (not the Dalvik heap).
c) Error codes are handled by wrapping them as exceptions and passing them up to Java through the NDK/JNI layer. Java is expected to handle the error properly.

What to Take Away from the Demo

Below is a brief description of each demo and instructions on how to work with each feature. All of the following demo were built with Android 2.2 as its Project build Target and can run on either an Android ARM device or an Android Virtual Device.

1. demo_view
Function: View a PDF page.
Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf
Generated files: None
After running the demo, you will see the following interface:



2. demo_view_navigation

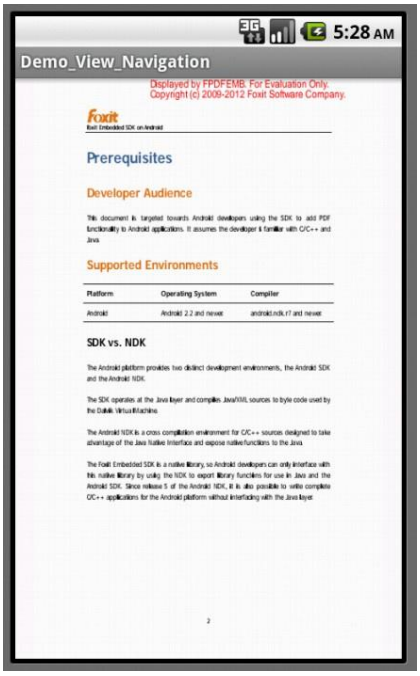
Features: Navigate next and pervious pages.

File used: readme.pdf and DroidSansFallback.ttf

Generated files: None

After running the demo, you will see something similar to the following interface:

2.1. Open the menu and select "Next Page" to go to the next page. Page will turn to the next page, as long as it is not the last page.



2.2. Open the menu and select "Previous Page" to go to the previous page. Page will turn to the previous page, as long as it is not the first page.

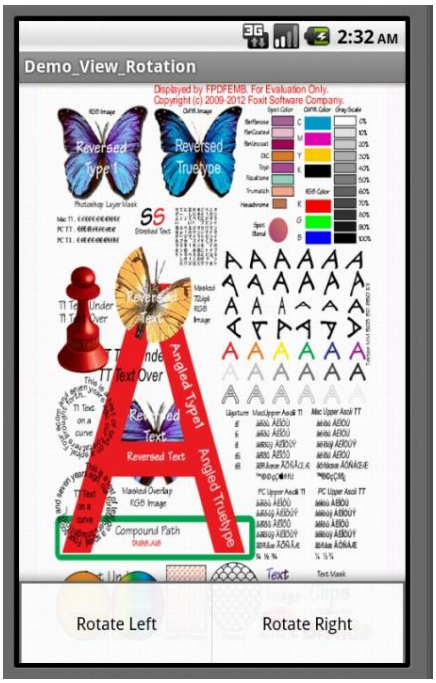
3. demo_view_rotation

Features: Rotate left and right.

Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

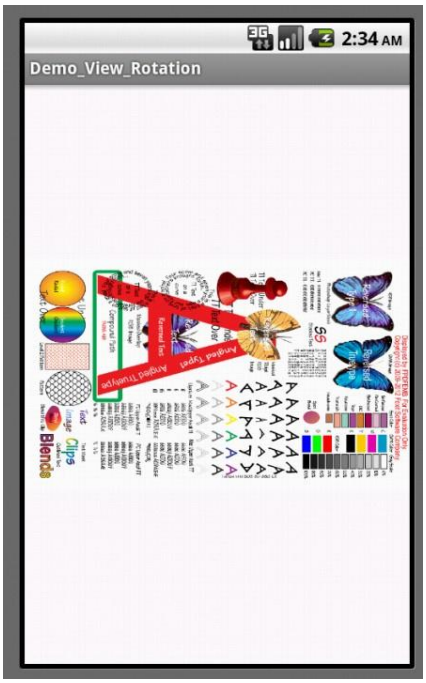
Generated files: None

After running the demo, you will see something similar to the following interface:



3.1. Open the menu and select "Rotate Left" to rotate the page 90 ° counter-clockwise.

3.2. Open the menu and select "Rotate Right" to rotate the page 90 ° clockwise.



4. demo_view_scrolling

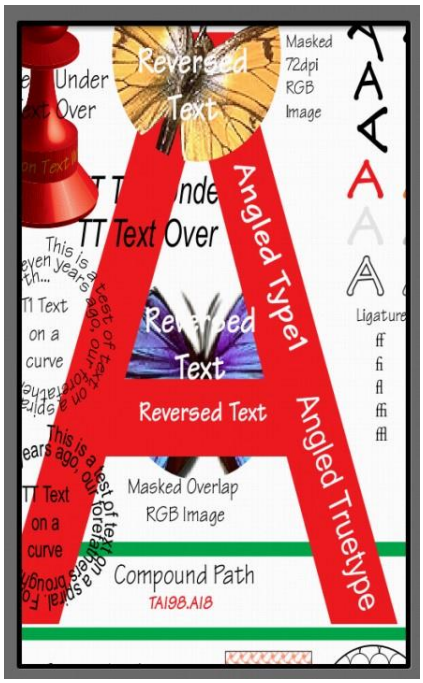
Features: Scroll.

Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

Generated files: None

After running the demo, you will see something similar to the following interface:

4.1. demo_view_scrolling supports touch scrolling. If you are using an Android ARM device, touch and drag will scroll the page. If you using Android Virtual device, click and drag will scroll the page.



5. demo_view_zoom

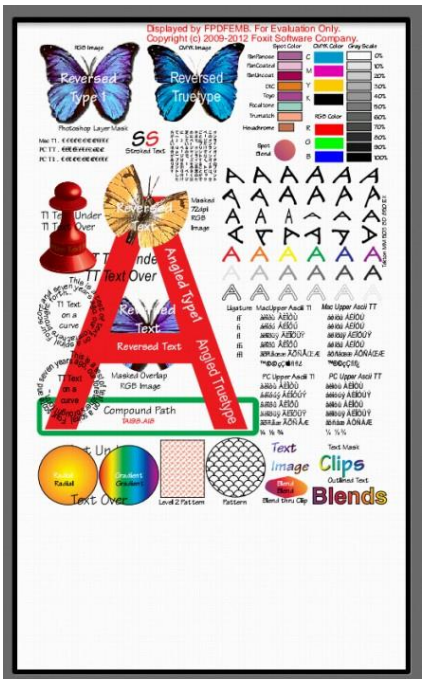
Features: Zoom to Fit Width, Zoom to fit Height, Zoom to Actual Size, Zoom in, and Zoom out.

Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

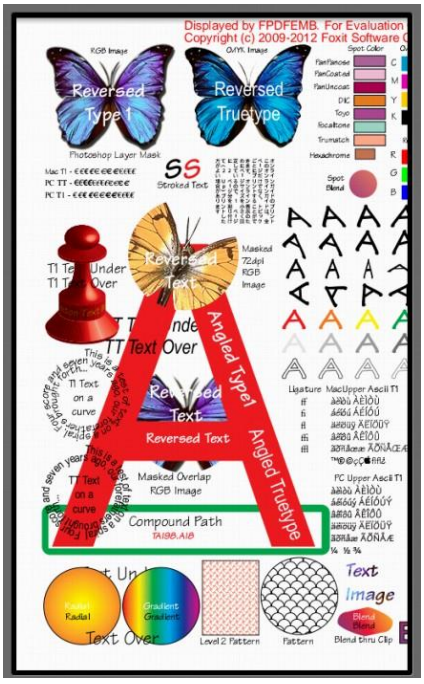
Generated files: None

After running the demo, you will see something similar to the following interface:

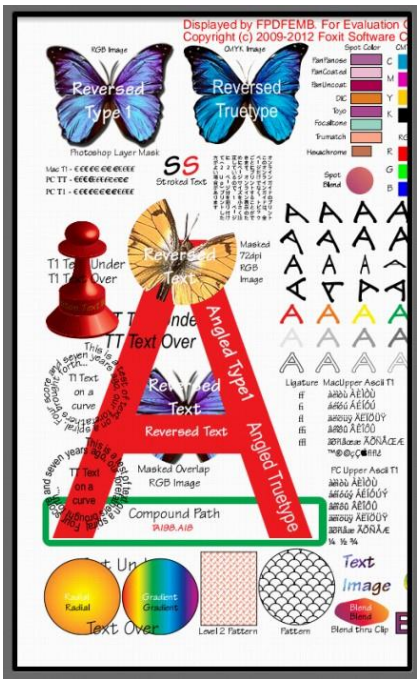
5.1 Open the menu and select "Fit Width" to fit the page in the display's width.



5.2 Open the menu and select "Fit Height" to fit the page in the display's height.

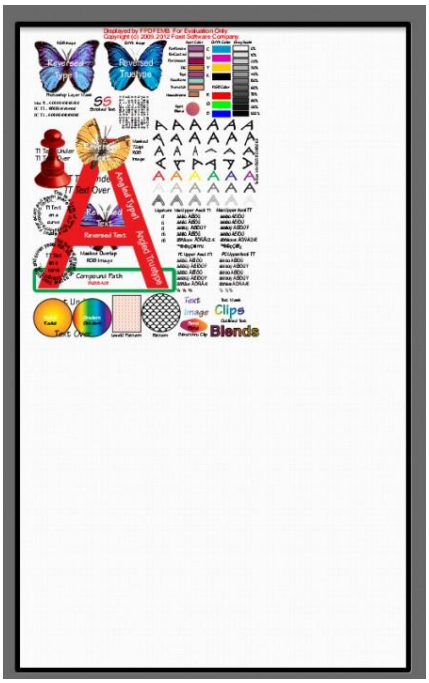


5.3 Open the menu and select "Actual Size" to display the page's original size.



5.4 Open the menu and select "Zoom In" to zoom in on the page by increase the horizontal and vertical dimensions.

5.5. Open the menu and select "Zoom out" to zoom out on the page by decreasing the horizontal and vertical dimensions.



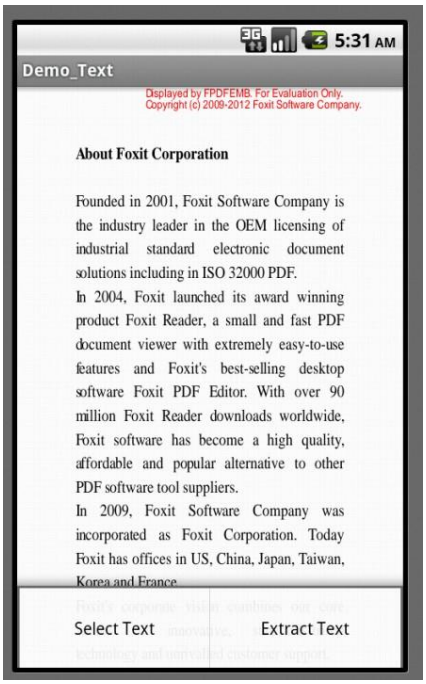
6. demo_text

Functions: Select and Extract Text.

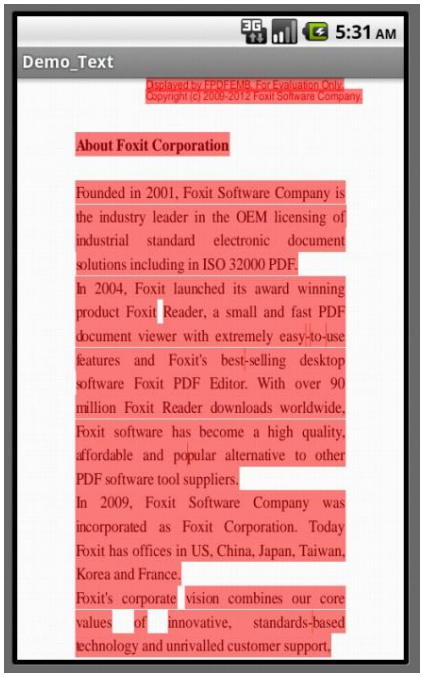
Files used: FoxitText.pdf and DroidSansFallback.ttf

Generated files: None

After running the demo, you will see something similar to the following interface:



6.1. Open the menu and select "Select Text" to select and highlight the text.



6.2. After the text has been selected, open the menu and select on "Extract Text" to display the text to the screen.

7. demo_search

Functions: Search text.

Files used: FoxitText.pdf and DroidSansFallback.ttf

Generated files: None

After running the demo, you will see something similar to the following interface:

7.1. Open the menu and select "Search" to search and highlight the first matching text.
Note: ”Foxit”was set as the text to be searched.

7.2. Open the menu and select “Next” to highlight the next occurrence of Foxit.

7.3. Open the menu and select "Previous" to highlight the previous occurrence of Foxit.

8. demo_annotations

Features: Add and Delete Annotations.

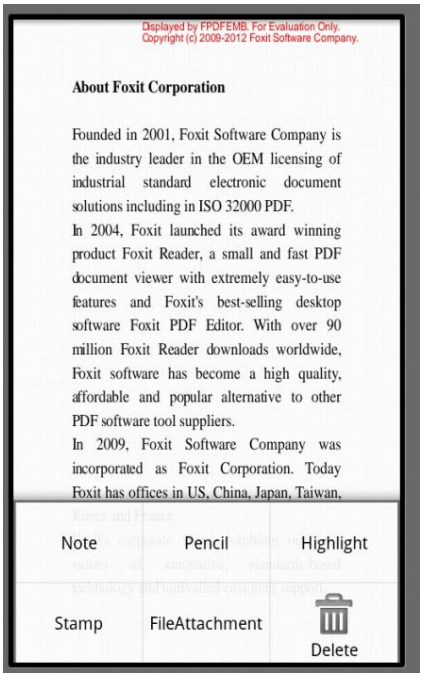
File used: FoxitText.pdf, DroidSansFallback.ttf, FoxitLogo.jpg, and FoxitForm.pdf

Generated file: FoxitSaveAnnotation.pdf

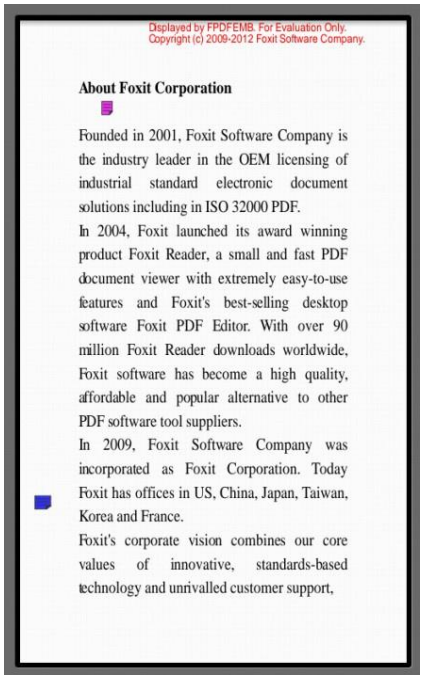
Attention: For demonstration purpose, an annotation can be added once and deleted once. All annotations are set to a fixed position in the demo. To show how to save an annotation

to PDF, the annotation demo automatically saves a copy of the PDF to /data/data/com.foxitsample.annotations/FoxitSaveAnnotation.pdf.

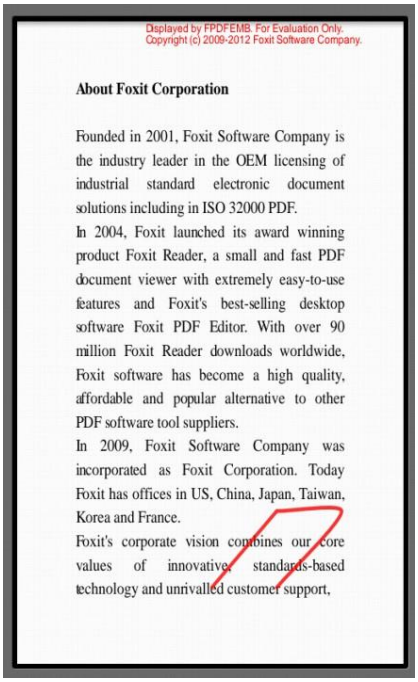
After running the demo, you will see something similar to the following interface:



8.1. Open the menu and select "Note" to add a Note annotation.

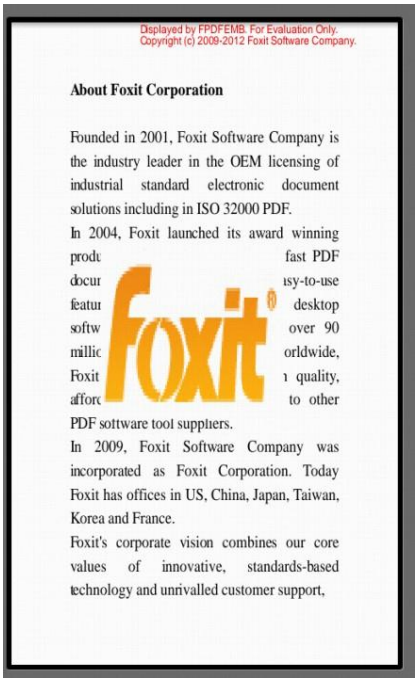


8.2 Open the menu and select "Pencil" to draw a pencil line annotation.

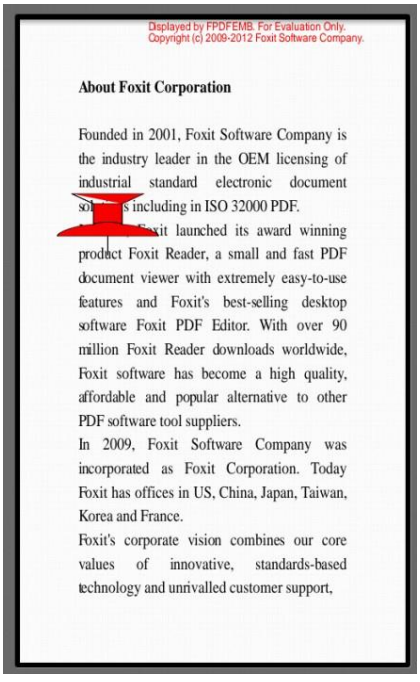


8.3. Open the menu and select "Hightlight" to add a highlight annotation.

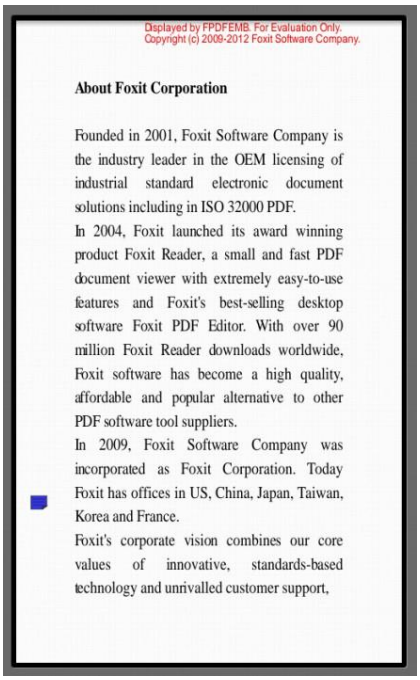
8.4. Open the menu and select "Stamp" to add a stamp annotation that contains the Foxit Logo graphics.



8.5. Open the menu and select "FileAttachment" to add a file attachment.

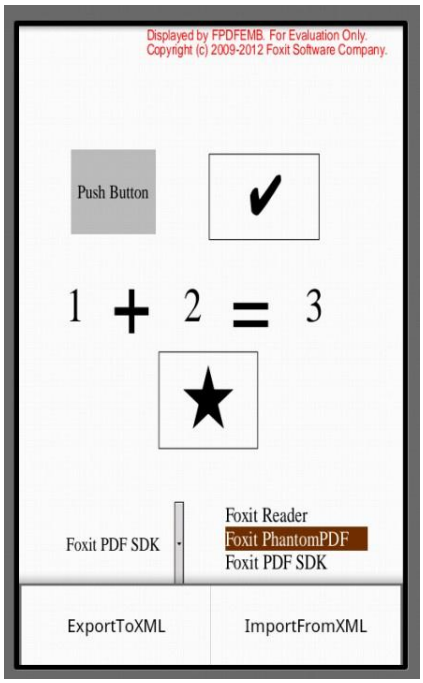


8.6. Open the menu and select "Delete" to delete any previously added annotation.

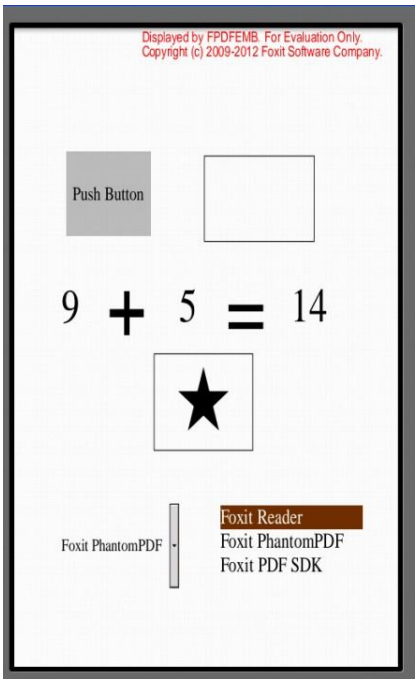


9. demo_form_field

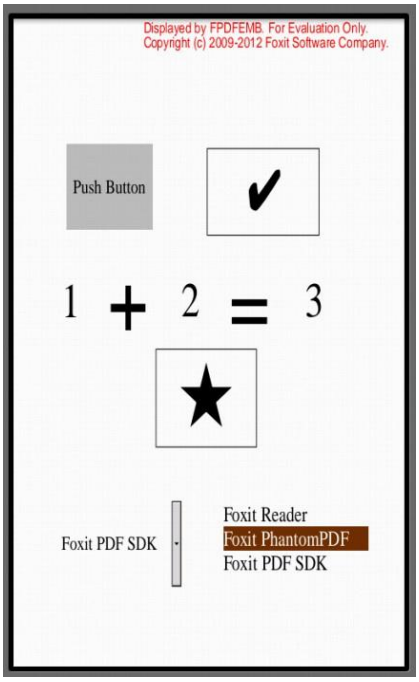
Functions: Fill in PDF forms, import Form data from xml, and export to PDF data to xml.
File used: FoxitForm.pdf and DroidSansFallback.ttf
Generated files: FormXml.xml
After running the demo, you will see something similar to the following interface:



9.1. Open the menu and select "ExportToXML" to export the form information to "/data/data/com.foxitsample.formfiled/FormXml.xml". After exporting is completed, make changes to the form data by selecting the data and changing the values. For example, you can make changes similar to what you see in the following screen shot.



9.2. After making changes, changes to the forms fields, Open the menu and select "ImportXML" to import the previously exported form data.



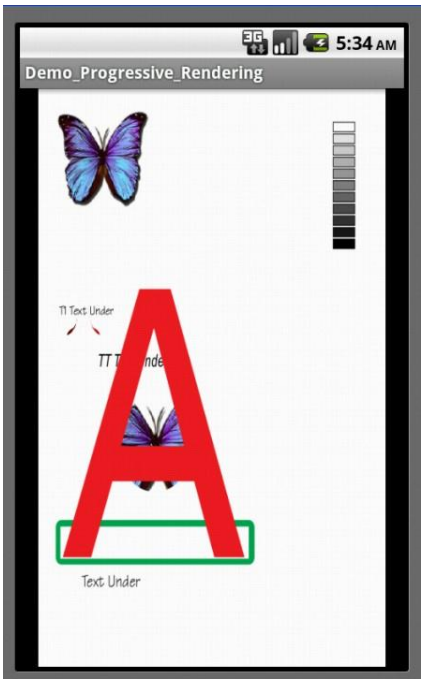
10. demo_progressive_rendering

Features: Pause and resume rendering.

Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

Generated files: None

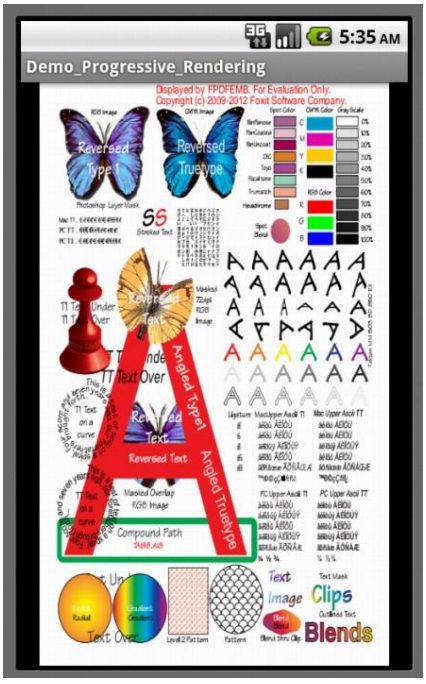
After running the demo, you will see the following interface:



10.1 For every 200ms, the demo will render a part of the page. The following screen shot shows what was render after 200ms.

10.2 The demo will continue to render.

10.3. The render will continue until the PDF is fully rendered.



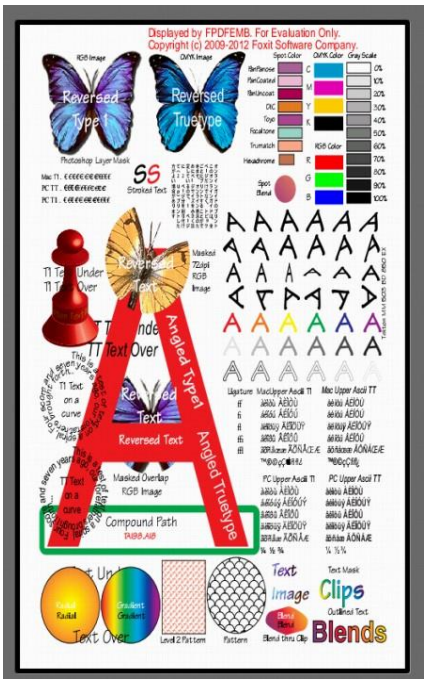
11. demo_psi

Functions: Add Pressure Sensitive Ink (PSI)

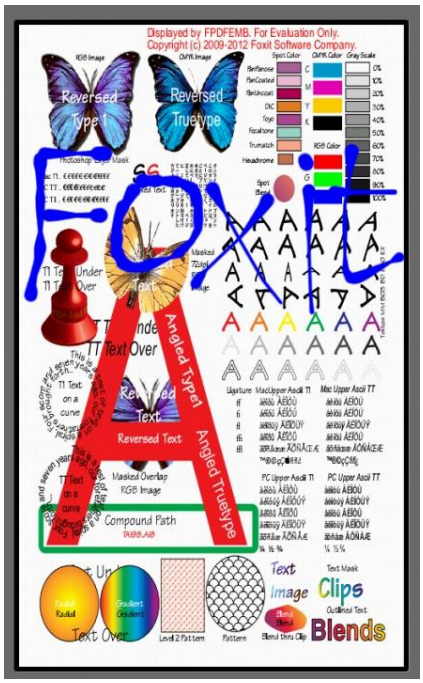
Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

Generated files: None

After running the demo, you will see the following interface:



11.1 Touch the screen (Android ARM device) or Click on the screen (Android Virtual device) to add a pressure sensitive ink.



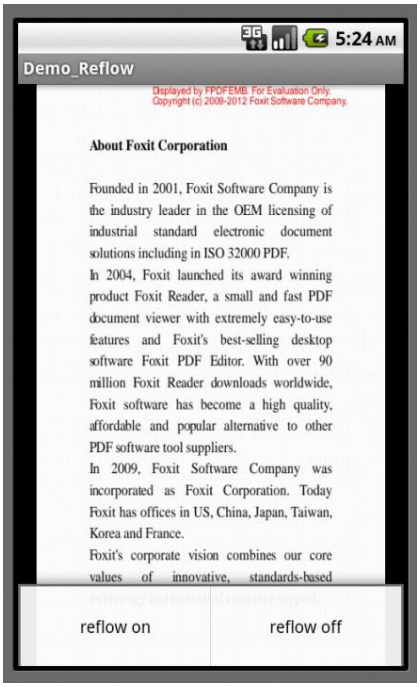
12. demo_reflow

Function: Change the arrangement of the page objects.

Files used: FoxitText.pdf and DroidSansFallback.ttf

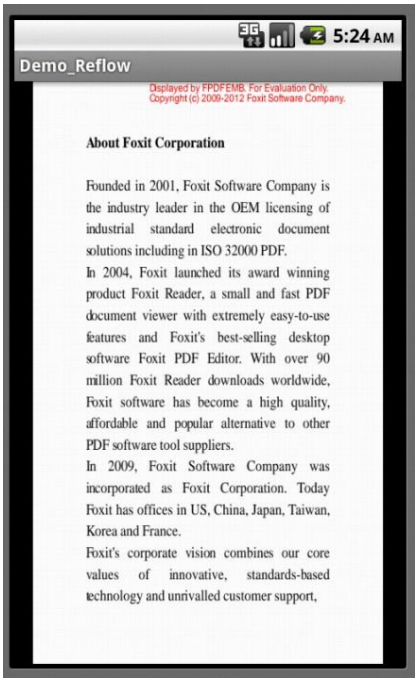
Generated files: None

After running the demo, you will see something similar to the following interface:



12.1. Open the menu and select "reflow on" to enable reflow.

12.2. Open menu and select "reflow off" to disable reflow.



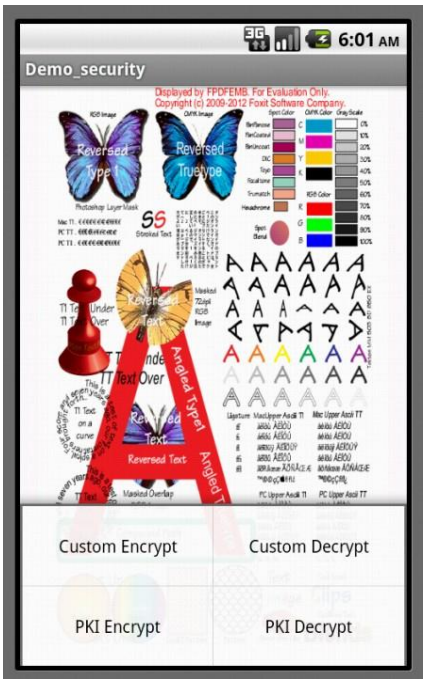
13. demo_security

Function: Encrypt and decrypt with Custom and PKI security.

Files used: FoxitBigPreview.pdf and DroidSansFallback.ttf

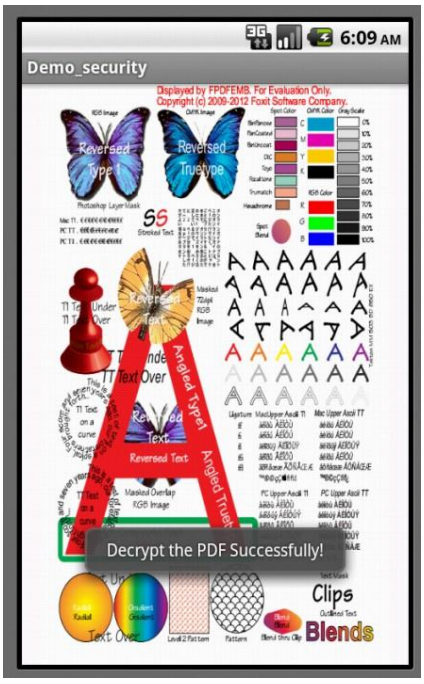
Generated files: FoxitEncryptd.pdf, FoxitDecryptd.pdf, FoxitPKIEncryptd.pdf, and FoxitPKIDecryptd.pdf

After running the demo, you will see something similar to the following interface:



13.1. Open and select "Custom Encrypt" to add a encrypt FoxitBigPreview.pdf to "data/data/com.foxitsample.security/FoxitEncryptd.pdf"

13.2. Open the menu and select "Custom Decrypt" to decrypt FoxitEncryptd.pdf to "/data/data/com.foxitsample.secruity/FoxitDecryptd.pdf" and to display the decrypted PDF.



13.3. PKI encryption will fail because envelope data and key data are invalid. Users are supposed to get envelope and key by themselves.