year 1
2010
project

Android PDF Writer

Description

Android PDF Writer (APW) is a simple Java library to generate simple PDF documents in Google's Android devices released under the BSD license.

The project page is: http://coderesearchlabs.com/androidpdfwriter

To learn more about how to use it, check the Usage section of this documentation.

There is a twin project for RIM's BlackBerry devices: the BlackBerry_PDF_Writer_(BPW) with the same features than this one. Also, both versions has minimal dependencies and none of them outside Java and System libraries:

- for Android:
  - java.util.ArrayList (in Body, Pages and List)
- for BlackBerry:
  - java.util.Vector (in Body, Pages and List)
  - net.rim.device.api.util.NumberUtilities (in CrossReferenceTable)

NOTE:
I wrote this library because I couldn't find a library to write PDFs in Android and I needed one to complete a weekend project.
So, the current APW library was coded in a weekend (May 14-16, 2010) following the PDF 1.4 Reference (I think my implementation is compatible with lower PDF versions also), so expect limited functionality: writes text, paint lines and rectangles... well, it covers pretty well the needs that my weekend project had at least, and may be it also address the needs of many other projects out there with simple reporting needs.

UPDATES:
Six months after that (Nov 15-16, 2010) I'm adding two things: first an overload to setPageFont() with an encoding parameter due to a user requirement for writing special characters in text. Check the PDF 1.4 Reference APPENDIX D, to be sure which encoding fits better for your needs, since as you can see there are symbols that have no octal value in all of the encodings. And second, an overload to addText() with a text transformation parameter, thanks to Jon from Gigaram Technologies for the suggestion.

By April 2011, I ported it to BlackBerry. And by June 2011, I added multiple pages support (thanks to Mark Heilpern for directing my attention back to this important feature). Implementing this, the methods setPageHeight() and setPageWidth() were removed. Instead, use the overloaded constructor of PDFWriter that accepts those parameters.

Finally, there are so many things that can be added (images, filters, etc) and enhanced (it does not handle key/values for instance, etc etc etc), so if you add something that is useful for you (I think this library is a good base to be extended, check the class structure), please let me know since it might be helpful for others too.

Enjoy,
Javier Santo Domingo
j-a-s-d@coderesearchlabs.com

Usage

APW is very simple to use, it offers an very straightfoward class to give you full control of the simple PDF document you are creating.

Look at this example:

```
private String generateHelloWorldPDF()
{
   PDFWriter mPDFWriter = new PDFWriter(400, 320);
   mPDFWriter.setPageFont(
      crl.android.pdfwriter.StandardFonts.SUBTYPE,
      crl.android.pdfwriter.StandardFonts.TIMES_ROMAN,
      crl.android.pdfwriter.StandardFonts.WIN_ANSI_ENCODING
   );
   mPDFWriter.addRawContent("1 0 0 rg\n");
   mPDFWriter.addText(70, 50, 12, "hello world");
   mPDFWriter.addRawContent("0 0 0 rg\n");
   mPDFWriter.addText(30, 90, 10, "© CRL", crl.android.pdfwriter.
   StandardFonts.DEGREES_270_ROTATION);
   mPDFWriter.addRawContent("[] 0 d\n");
   mPDFWriter.addRawContent("1 w\n");
   mPDFWriter.addRawContent("0 0 1 RG\n");
   mPDFWriter.addRawContent("0 1 0 rg\n");
   mPDFWriter.newPage();
   mPDFWriter.addRectangle(40, 50, 280, 50);
   mPDFWriter.addText(85, 75, 18, "Code Research Laboratories");
   mPDFWriter.newPage();
   mPDFWriter.setPageFont(
      crl.android.pdfwriter.StandardFonts.SUBTYPE,
      crl.android.pdfwriter.StandardFonts.COURIER_BOLD
   );
   mPDFWriter.addText(150, 150, 14, "http://coderesearchlabs.com");
   mPDFWriter.addLine(150, 140, 270, 140);
   String s = mPDFWriter.asString();
   return s;
}
```

As you can see, it is strictly a writer where you have to write pages and content sequentially.

To learn more about the library, check the The Library section of this documentation.

The Library

The main class of this library is **PDFWriter**, which presents the following public members:

- `PDFWriter()`
- `PDFWriter(int pageHeight, int pageWidth)`
- `String asString()`
- `void newPage()`
- `void setPageFont(String subType, String baseFont)`
- `void setPageFont(String subType, String baseFont, String encoding)`
- `void addRawContent(String rawContent)`
- `void addText(int leftPosition, int topPositionFromBottom, int fontSize, String text)`
- `void addText(int leftPosition, int topPositionFromBottom, int fontSize, String text, String transformation)`
- `void addLine(int fromLeft, int fromBottom, int toLeft, int toBottom)`
- `void addRectangle(int fromLeft, int fromBottom, int toLeft, int toBottom)`

All the font and text options are included in the **StandardFonts** class, presenting the following variety of public members.

Subtype constants:

- `String SUBTYPE`

Font constants:

- `String TIMES_ROMAN`
- `String TIMES_BOLD`
- `String TIMES_ITALIC`
- `String TIMES_BOLDITALIC`
- `String HELVETICA`
- `String HELVETICA_BOLD`
- `String HELVETICA_OBLIQUE`
- `String HELVETICA_BOLDOBLIQUE`
- `String COURIER`
- `String COURIER_BOLD`
- `String COURIER_OBLIQUE`
- `String COURIER_BOLDOBLIQUE`
- `String SYMBOL`
- `String ZAPDINGBATS`

Encoding constants:

- `String MAC_ROMAN_ENCODING`
- `String WIN_ANSI_ENCODING`

Rotation constants:

- `String DEGREES_0_ROTATION`
- `String DEGREES_45_ROTATION`
- `String DEGREES_90_ROTATION`
- `String DEGREES_135_ROTATION`
- `String DEGREES_180_ROTATION`
- `String DEGRESS_225_ROTATION`
- `String DEGREES_270_ROTATION`
- `String DEGREES_315_ROTATION`

License

Tools

In the development of **Android PDF Writer** the following tools were specifically involved:

for source code editing
Eclipse Foundation's Eclipse
http://www.eclipse.org

for source code compiling
Google's Android SDK
http://developer.android.com/sdk

When working at **Code Research Laboratories** the following tools are used:

in the test management field
Gurock Software's TestRail
http://www.gurock.com/testrail/

in the version control field
VisualSVN Ltd's VisualSVN
http://www.visualsvn.com/

in the similarity analysing field
RedHill Consulting's Simian
http://www.redhillconsulting.com.au/products/simian/

in the application lifecycle management field
Inedo's BuildMaster
http://www.inedo.com/buildmaster/

in the documentation field
EC Software's Help & Manual
http://www.ec-software.com/products_hm_overview.html

History

DATE	DESCRIPTION
2010.05.14	started coding
2010.05.15	improvements
2010.05.16	more improvements
2010.05.19	initial release
2010.11.15	added setPageFont() overload with encoding parameter
2010.11.16	added addText() overload with text transformation
2011.04.20	ported to BlackBerry (creating the BlackBerry PDF Writer twin project)
2011.04.25	added addRectangle()
2011.05.19	added Documentation
2011.06.15	added multiple pages support