Developer's Guide to Asprise Java PDF Writer/Reader Library

Multi-platforms

Last updated on March 30, 2007

Table of Contents

1 Introduction

Asprise offers Java PDF writer and reader library as valued add-on to our flagship products – Asprise OCR & JTwain. With this library, you can create PDF files from separate images easily and vice verse. Additionally, you can use the library to extract text from PDF files.

Download Asprise PDF library from: http://asprise.com/product/javapdf

The evaluation kit contains the following files:

File/Dir	Remarks
javadoc	Contains the Javadoc API for Asprise PDF Writer/Reader classes
AspriseJavaPDF.jar	Contains Java PDF Writer/Reader related classes;
img/girl.jpg	Sample image 1
img/sky.jpg	Sample image 2
demo-write.bat/sh	Demo program. Double click to run it – this demo creates test.pdf from the two sample images.
demo-read.bat/sh	Demo program. Double click to run it – this demo extract text content from test.pdf – you will not see any text extracted as the PDF file contains images only. Modify demo-read.bat/sh to extract text from your own PDF files.

Trying them will give you some feelings about the Java PDF writer and reader.

2 PDF Writer

Main class: **com.asprise.util.pdf. PDFImageWriter**

You can use this class to create PDF files from images.

2.1 Sample Program

```
1.  PDFImageWriter writer = new PDFImageWriter(new
    FileOutputStream("new.pdf"));
2.   writer.open();
3.   writer.addImage("C:\\1.jpg");
4.   writer.addImage("C:\\2.png");
5.   writer.close();
6.   System.out.println("DONE.");
```

First you create a PDFImageWriter (Line 1). Before adding content to the PDF, you need to open() it (Line 2). Then you can add images to it – simply pass image paths. Supported image formats are: GIF, JPEG, BMP, PNG. Finally, you finish write the PDF by closing it (Line 5).

In fact, there is an even easier way to create the PDF file:

```
PDFImageWriter.saveImagesToPDF(new File[]{ new File("C:\\1.jpg"), new
File("C:\\2.png") }, new File("new.pdf");
```

2.2 Main Functions

2.2.1 addImage

addImage(java.lang.String imageFilePath)

addImage(java.io.File imageFile)

addImage(java.awt.Image image)

You use addImage functions to insert images into the output PDF file.

The source image can be an image file or a standard Java image (java.awt.Image). The following image file formats are directly supported:

- BMP
- GIF
- JPEG
- PGN

In case you need convert images from file formats other than above types, you can use third party library to load the image into java.awt.Image and then use addImage(java.awt.Image) to add to the PDF.

For TIFF support, you can use Asprise TIFF library http://asprise.com/product/javatiff

2.2.2 saveImagesToPDF

saveImagesToPDF(java.io.File[] sourceImages, java.io.File targetPDFile)

saveImagesToPDF(java.awt.Image[] sourceImages, java.io.File targetPDFile)

We strive to make developers' life easier. With saveImagesToPDF methods, you can create PDF files from multiple images with a single line code:

```
PDFImageWriter.saveImagesToPDF(new File[]{ new File("C:\\1.jpg"), new File("C:\\2.png") }, new File("new.pdf");
```

Alternatively, you can create a PDF file from multiple java.awt.Image objects.

2.3 PDF Security/Encryption

Asprise PDF library allow you to add security restriction to your PDF files.

For example, the code demonstrates the usage of Asprise writer for PDF encryption:

```
1.  PDFSecurityObject so = new PDFSecurityObject();
2.   so.ownerPassword = "LAB Asprise";
3.   so.userPassword = "user";
4.   so.permissions = PDFSecurityObject.PERMISSION_PRINTING |
    PDFSecurityObject.PERMISSION_COPY; // allow printing and copying.
5.
6.   PDFImageWriter writer = new PDFImageWriter(new
    FileOutputStream("new.pdf"));
7.   writer.setPDFSecurityObject(so);
8.   writer.open();
9.   writer.addImage("C:\\1.jpg");
10.  writer.close();
```

PDFSecurityObject class is used for security settings (Line 1), you use it to instruct the PDFImageWriter to add security (Line 7) to the output PDF file.

In above example, we set both the owner password and user password. When a user opens the PDF file, he or she will be prompted to enter the user password. If the user successfully enter the password, then the user can view and print (Line 4) the PDF.

For more details on security settings, please view the javadoc of the PDFSecurityObject class.

3 PDF Reader

Asprise PDF reader can be used to **extract text from PDF pages**, and rasterize PDF content to images.

3.1 Sample Program

```
1.  PDFReader reader = new PDFReader(new File("my.pdf"));
2.   reader.open(); // open the file.
3.   int pages = reader.getNumberOfPages();
4.
5.   for(int i=0; i<pages; i++) {
6.     String text = reader.extractTextFromPage(i);
7.    System.out.println("Page " + i + ": " + text);
8.   }
9.
10.  ... // perform other operations on pages.
11.
12.  reader.close(); // finally, close the file.
```

First, a PDFReader is created from an existing PDF file. The open() function in Line 2 will actually loads the PDF file. You use getNumberOfPages function to enquire the total number of pages in the PDF file (Line 3). Then you can process each page. Finally, you need to close (i.e., unload) the document as in Line 12.

3.2 Main Functions

3.2.1 getNumberOfPages

Returns the total number of pages in the PDF file.

3.2.2 getPageSize(pageIndex)

The getPageSize function returns the dimension of the specified page. It is common that a PDF has pages with different dimensions.

3.2.3 extractTextFromPage(pageIndex)

The extractTextFromPage function extracts text content from the specified page and returns the text as a string.

The text extracted can be used for indexing. For example, you can feed the text to Apache Lucene engine (http://lucene.apache.org) to make PDF files searchable.

3.2.4 getPageAsImage(pageIndex)

The getPageAsImage rasterize the specified PDF page and returns the result as a java.awt.image.BufferedImage.

For the result image, you can feed it to Asprise OCR engine for text recognition http://asprise.com/product/ocr

For more details, please view the Javadoc of Asprise PDF library.

3.3 PDF Security/Encryption

PDFReader allows you to load a PDF with encryption. Below describes the steps to read such a PDF file:

1. Create a PDFReader object;
2. Create a PDFSecurityObject object and set the user password;
3. Call PDFReader.setSecurityObject to associate the PDFSecurityObject with the reader;
4. Then open() the reader and you can process the pages.

4 License Schemes

Asprise PDF library has the lowest ownership cost and we offer flexible licensing schemes.

License	Terms	Price
Writer Only	Royalty-free distribution – distribute to unlimited number of computers/CPU	USD 398/developer
Reader Only	Royalty-free distribution – distribute to unlimited number of computers/CPU	USD 998/developer
Writer & Reader	Royalty-free distribution – distribute to unlimited number of computers/CPU	USD 1298/developer
	Royalty-free distribution + Site License	USD 2498/site
	Royalty-free distribution + Site License + Full Source Code	USD 3998/site

The above prices are subjected to change. Please visit http://asprise.com/product/javapdf/order.php for latest price list and order instructions.