FOP: Fonts

$Revision: 326073 $

by Jeremias Märki, Tore Engvig

Table of contents

1. Summary

The following table summarizes the font capabilities of the various FOP renderers:

Renderer	Base-14	AWT/OS	Custom	Custom Embedding
PDF	yes	no	yes	yes
PostScript	yes	no	yes	no
PCL	yes (modified)	no	no	no
TXT	yes (used for layout but not for output)	no	yes (used for layout but not for output)	no
AWT	if available from OS	yes	yes	n/a (display only)
Print	if available from OS	yes	yes	controlled by OS printer driver
RTF	n/a (font metrics not needed)	n/a	n/a	n/a
MIF	n/a (font metrics not needed)	n/a	n/a	n/a
SVG	if available from OS	yes	no	no
XML	yes	no	yes	n/a

2. Base-14 Fonts

The Adobe PDF Specification specifies a set of 14 fonts that must be available to every PDF reader: Helvetica (normal, bold, italic, bold italic), Times (normal, bold, italic, bold italic), Courier (normal, bold, italic, bold italic), Symbol and ZapfDingbats.

3. AWT/Operating System Fonts

The AWT family of renderers (AWT, Print, SVG), use the Java AWT libraries for font metric information. Through operating system registration, the AWT libraries know what fonts are

available on the system, and the font metrics for each one.

4. Custom Fonts

Support for custom fonts is added by creating font metric files (written in XML) from the actual font files, and registering them with FOP. Currently only Type 1 and TrueType fonts can be added. More information about fonts can be found at:

- Adobe font types
- Adobe Font Technote

4.1. Type 1 Font Metrics

FOP includes PFMReader, which reads the PFM file that normally comes with a Type 1 font, and generates an appropriate font metrics file for it. To use it, run the class org.apache.fop.fonts.apps.PFMReader:

Windows:

```
java -cp build\fop.jar;lib\avalon-framework.jar;lib\xml-apis.jar;
         lib\xercesImpl.jar;lib\xalan.jar
           org.apache.fop.fonts.apps.PFMReader [options] pfm-file xml-file
```

Unix:

```
java -cp build/fop.jar:lib/avalon-framework.jar:lib/xml-apis.jar:
         lib/xercesImpl.jar:lib/xalan.jar
           org.apache.fop.fonts.apps.PFMReader [options] pfm-file xml-file
```

PFMReader [options]:

- **-fn <fontname>** By default, FOP uses the fontname from the .pfm file when embedding the font. Use the "-fn" option to override this name with one you have chosen. This may be useful in some cases to ensure that applications using the output document (Acrobat Reader for example) use the embedded font instead of a local font with the same name.

Note:

The classpath in the above example has been simplified for readability. You will have to adjust the classpath to the names of the actual JAR files in the lib directory. avalon-framework.jar is necessary only for versions 0.20.5 or later. xml-apis.jar, xercesImpl.jar and xalan.jar are not necessary for JDK version 1.4 or later.

Note:

The tool will construct some values (FontBBox, StemV and ItalicAngle) based on assumptions and calculations which are only an approximation to the real values. FontBBox and Italic Angle can be found in the human-readable part of the PFB file or in the AFM file. The PFMReader tool does not yet interpret PFB or AFM files, so if you want to be correct, you may have to adjust the values in the XML file manually. The constructed values however appear to have no visible influence.

4.2. TrueType Font Metrics

FOP includes TTFReader, which reads the TTF file and generates an appropriate font metrics file for it. Use it in a similar manner to PFMReader. For example, to create such a metrics file in Windows from the TrueType font at c:\myfonts\cmr10.ttf:

```
java -cp build\fop.jar;lib\avalon-framework.jar;lib\xml-apis.jar;
         lib\xercesImpl.jar;lib\xalan.jar
           org.apache.fop.fonts.apps.TTFReader [options]
             C:\myfonts\cmr10.ttf ttfcm.xml
```

TTFReader [options]:

- **-d <DEBUG | INFO >** Sets the debug level (default is INFO).
- **-fn <fontname>** Same as for PFMReader.
- **-ttcname <fontname>** If you're reading data from a TrueType Collection (.ttc file) you must specify which font from the collection you will read metrics from. If you read from a .ttc file without this option, the fontnames will be listed for you.
- **-enc ansi** Creates a WinAnsi-encoded font metrics file. Without this option, a CID-keyed font metrics file is created. The table below summarizes the differences between these two encoding options as currently used within FOP. Please note that this information only applies to TrueType fonts and TrueType collections:

Issue	WinAnsi	CID-keyed
Usable Character Set	Limited to WinAnsi character set, which is roughly equivalent to iso-8889-1.	Limited only by the characters in the font itself.
Character Encoding in the Output Document.	Correct.	Never correct. Search, index, and cut-and-paste operations in the output document will produce incorrect results.
Character Display	Correct.	Correct if and only if the font is embedded in the output. (This is possible because, although the underlying characters are encoded incorrectly, the embedded font is also encoded incorrectly).

Warning:

As shown in the above table, regardless of whether the font is embedded or not, text generated from a CID-keyed font metrics file will *never* be encoded properly. Further, if the related font is not embedded, it cannot even be displayed properly. Obviously, this behavior is not desirable, and we hope to correct it in upcoming releases.

4.3. TrueType Collections Font Metrics

TrueType collections (.ttc files) contain more than one font. To create metrics files for these fonts, you must specify which font in the collection should be generated, by using the "-ttcname" option with the TTFReader.

To get a list of the fonts in a collection, just start the TTFReader as if it were a normal TrueType file (without the -ttcname option). It will display all of the font names and exit with an Exception.

Here is an example of generating a metrics file for a .ttc file:

```
java -cp build\fop.jar;lib\avalon-framework.jar;lib\xml-apis.jar;
         lib\xercesImpl.jar;lib\xalan.jar
           org.apache.fop.fonts.apps.TTFReader -ttcname "MS Mincho"
             msmincho.ttc msminch.xml
```

4.4. Register Fonts with FOP

You must tell FOP how to find and use the font metrics files by registering them in the FOP Configuration. Add entries for your custom fonts, regardless of font type, to the configuration file in a manner similar to the following:

```
<font metrics-file="FTL_____.xml" kerning="yes"
      embed-file="C:\myfonts\FTL_____.pfb">
  <font-triplet name="FrutigerLight" style="normal" weight="normal"/>
</font>
```

Note:
Review the documentation for FOP Configuration for instructions on making the FOP configuration available to FOP when it runs. Otherwise, FOP has no way of finding your custom font information.

- Starting with FOP version 0.20.5 you can use URLs for the paths to the font files. Relative URLs are resolved relative to the fontBaseDir property (or baseDir) if available. See FOP: Configuration for more information.
- The "kerning" and "embed-file" attributes are optional. Kerning is currently not used at all. If embedding is off, the output will position the text correctly (from the metrics file), but it will not be displayed or printed correctly unless the viewer has the applicable font available to their local system.
- When setting the embed-file attribute for Type 1 fonts, be sure to specify the PFB (actual font data), not PFM (font metrics) file that you used to generate the XML font metrics file.

Note:
Cocoon users will need to setup the config, see FOPSerializer for more information.

4.5. Embedding

Note:
The PostScript renderer does not yet support font embedding.

Note:
The font is simply embedded into the PDF file, it is not converted.

Font embedding is enabled in the userconfig.xml file and controlled by the embed-file attribute. If you don't specify the embed-file attribute the font will not be embedded, but will only be referenced.

When FOP embeds a font, it adds a prefix to the fontname to ensure that the name will not match the fontname of an installed font. This is helpful with older versions of Acrobat Reader that preferred installed fonts over embedded fonts.

When embedding PostScript fonts, the entire font is always embedded.

When embedding TrueType fonts (ttf) or TrueType Collections (ttc), a subset of the original font, containing only the glyphs used, is embedded in the output document. Currently, this embedded font contains only the minimum data needed to be embedded in a pdf document, and does not contain any codepage information. The PDF document contains indexes to the glyphs in the font instead of to encoded characters. While the document will be displayed correctly, the net effect of this is that searching, indexing, and cut-and-paste will not work properly.

One workaround for this behavior is to use the "-enc ansi" option when generating metrics with TTFReader. This will cause the whole font to be embedded in the pdf document. Characters will be WinAnsi encoded (as specified in the PDF spec), so you lose the ability to use characters from other character sets. See Table of TTF Encoding Options for more details.