Using the Command Line Interface

by Jeremias Märki

Table of contents

1 Introduction..... 2
2 Features..... 2
3 Running the command line application..... 2
4 Parameters..... 2
4.1 Example 1..... 3
4.2 Example 2..... 3
4.3 Example 3..... 3
5 Tweaking the barcode settings..... 3

Copyright © 2003-2012 The Barcode4J Project. All rights reserved.

1 Introduction

This page describes how to use the command line interface that comes with **Barcode4J**.

2 Features

The command line interface has the following features:

- Output to the console or to a file
- Output formats: SVG, EPS, bitmap formats
- Ability to set the same symbology properties as in the XSLT or XSL-FO extensions (via a configuration file)
- Ability to set the resolution and color settings for the bitmaps

3 Running the command line application

The main class for the command line interface is `org.krysalis.barcode4j.cli.Main`. Either setup the classpath manually (-cp option) or use the barcode.bat batch file (Windows only).

Required libraries are:

- `barcode4j.jar` (from the `build/` directory)
- `avalon-framework-4.2.0.jar` (Apache Avalon Framework, from the `lib/` directory)
- `commons-cli-1.0.jar` (Apache Jakarta Commons CLI, from the `lib/` directory)

An example:

```
Windows:
java -cp build\barcode4j.jar;lib\avalon-framework-4.1.5.jar;lib\commons-cli-1.0.jar
    org.krysalis.barcode4j.cli.Main <parameters>

Unix:
java -cp build/barcode4j.jar:lib/avalon-framework-4.1.5.jar:lib/commons-cli-1.0.jar
    org.krysalis.barcode4j.cli.Main <parameters>
```

4 Parameters

Calling the Command Line Interface without parameters brings up the help screen:

```
Barcode4J command-line application, Version 2.0

usage: java -jar barcode4j.jar [-v] [[-s <symbology>]|[-c <cfg-file>]] [-f
            <format>] [-d <dpi>] [-bw] [-o <file>] <message>
    --bw                (for bitmaps) create monochrome (1-bit) image
                        instead of grayscale (8-bit)
 -s,--symbol <name>     the barcode symbology to select (default settings,
                        use -c if you want to customize)
 -c,--config <file>     the config file
 -d,--dpi <integer>     (for bitmaps) the image resolution in dpi
                        Default: 300
 -f,--format <format>   the output format: MIME type or file extension
                        Default: image/svg+xml (SVG)
 -o,--output <file>     the output filename
 -v,--verbose           enable debug output
```

Copyright © 2003-2012 The Barcode4J Project. All rights reserved.

```
Valid output formats:
SVG: image/svg+xml, svg
EPS: image/x-eps, eps
PNG: image/x-png, png
TIFF: image/tiff, tiff, tif (unavailable)
JPEG: image/jpeg, jpeg, jpg
GIF: image/gif, gif (unavailable)
If -o is omitted the output is written to stdout.
```

> **Note:**
> You'll notive the "(unavailable)" notice above. Not all image formats are available in every environment.

Hopefully, the above is self-explanatory, so we just provide a few examples.

4.1 Example 1

```
java -cp <classpath> org...cli.Main -s code128 "MyMessage"
```

This creates SVG output (because SVG is default if no format is given). The SVG is written to the console due to the absence of an output filename. The symbology used is "Code 128" (-s code128).

4.2 Example 2

```
java -cp <classpath> org...cli.Main -o barcode.eps -f eps -s ean13 "008888650997"
```

This creates an EAN-13 (-f ean13) barcode that is written to a file (-o) in EPS format (-f eps).

4.3 Example 3

```
java -cp <classpath> org...cli.Main -o barcode.png -f png --bw -d 600 -s ean13 "008888650997"
```

This creates the same barcode as above but generates a monochrome (--bw) PNG bitmap (-f png) instead. The resolution is set to 600dpi (-d 600).

5 Tweaking the barcode settings

Often, you will need to change the barcode settings because the default values don't provide the desired result. In this case you have to provide a little XML file with the configuration for the barcode symbology. The XML format used there is again the same Barcode XML as used everywhere else.

Here's an example configuration file (for example: barcode-cfg.xml):

```
<?xml version="1.0" encoding="UTF-8"?>
<barcode>
  <code128>
    <height>2cm</height>
    <module-width>0.5mm</module-width>
  </code128>
</barcode>
```

Copyright © 2003-2012 The Barcode4J Project. All rights reserved.

The above sets up a Code 128 barcode. To generate the barcode you use the -c parameter instead of the -s parameter. The first example again, this time with a -c parameter:

```
java -cp <classpath> org...cli.Main -c barcode-cfg.xml "MyMessage"
```

Copyright © 2003-2012 The Barcode4J Project. All rights reserved.