Frequently Asked Questions

Table of contents

Questions

1. Problems

1.1. My barcode scanner can't read the barcode.

There are many things that can go wrong when you generate and print barcodes. Please go to the Troubleshooting page for more information.

1.2. My barcode scanner occasionally sends incorrect characters.

Please consult the Troubleshooting page for more information.

2. Documentation

2.1. How can I help write documentation?

This project uses Apache Forrest to generate documentation from XML. Please download a copy of Forrest, which can be used to validate, develop and render a project site.

3. Apache FOP

3.1. The FOP extension fails.

You will see error messages like this:

```
[ERROR] Unknown formatting object http://barcode4j.krysalis.org/ns^barcode
[ERROR] no handler defined for http://barcode4j.krysalis.org/ns:barcode
foreign xml
```

This usually indicates that the FOP extension is not present in the classpath. Make sure you've got either both barcode4j.jar and barcode4j-fop-ext-0.20.5.jar or the bigger barcode4j-fop-ext-0.20.5-complete.jar in your classpath. If you work from the command-line you may have to change fop.bat or fop.sh.

In rare circumstances this may be a result of a classloader problem. The resolution of problems like this depends on your environment.

3.2. I've got error messages that contain the words "avalon" and "framework". What's wrong?

Most probably you have an outdated version of Avalon Framework in your classpath. This can happen if you're working with FOP 0.20.5, for example, which contains an older version of Avalon Framework. Please use the one supplied with Barcode4J or download the latest release directly from the Avalon site.

4. Java

4.1. What's this "classpath" thing?

The classpath is a set of locations where the Java Virtual Machine (JVM) can load Java code from. In the easiest case, the classpath is simply a directory, but normally the classpath consists of a list of filenames pointing to JAR files (essentially ZIP files with some special extensions). Barcode4J delivers several such JAR files. Let's see what the classpath looks like when you want to run Barcode4J from the command-line:

```
Windows:
java -cp
lib\avalon-framework-4.2.0.jar;lib\commons-cli-1.0.jar;build\barcode4j.jar
    org.krysalis.barcode4j.cli.Main

Unix:
java -cp
lib\avalon-framework-4.2.0.jar:lib\commons-cli-1.0.jar:build\barcode4j.jar
    org.krysalis.barcode4j.cli.Main
```

Note:
Instead of "-cp" you can also use "-classpath".

The above works fine under JDK 1.4.x or higher. If you use JDK 1.3.x you need to add XML support (JAXP, Xerces and Xalan). JDK 1.4 has XML support built-in. You would add these libraries using the same pattern as shown above.

4.2. How do I make sure the right XML parser and the right XSLT engine are used under JDK 1.4.x and higher?

Starting with JDK 1.4, Java comes with XML support built-in. XML support comprises the JAXP API (SAX, DOM, parsing and XSLT processing APIs) plus implementations

(Crimson/Xerces-J for basic XML support and Xalan-J as XSLT engine).

By default, the implementations coming with the JDK are used even if you have different JAXP implementations in your classpath. There's a mechanism called "Endorsed Standards Override Mechanism" that allows you to override the default XML implementations with your own. See Xalan-J's FAQ entry or the documentation from Sun.

4.3. What do I need to do to run Barcode4J on a "headless" server?

Some output formats like EPS and SVG work on a headless server without any precautions, since they don't use any of the AWT infrastructure.

However, if you use AWT or bitmap output Barcode4J needs a functional graphical environment. Furthermore, if you generate SVG barcode in conjunction with Apache FOP you will need a graphical environment since FOP uses Batik to render the barcodes. Batik needs the graphical environment to work.

The work-arounds are the same as for Apache FOP:

- If you are using JDK 1.4 or later, start it with the command line option: `-Djava.awt.headless=true`
- Install an X server which provides an in-memory framebuffer without actually using a screen device or any display hardware. One example is Xvfb.
- Install a toolkit which emulates AWT without the need for an underlying X server. One example is the PJA toolkit, which is free (GPL license) and comes with detailed installation instructions.