Apache™ FOP: Hyphenation

$Revision$

Table of contents

1 Hyphenation Support....2
1.1 Introduction....2
1.2 License Issues....2
1.3 Sources of Custom Hyphenation Pattern Files....2
1.4 Installing Custom Hyphenation Patterns....2
2 Hyphenation Patterns....3

1 Hyphenation Support

1.1 Introduction

Apache™ FOP uses Liang's hyphenation algorithm, well known from TeX. It needs language specific pattern and other data for operation.

Because of licensing issues (and for convenience), all hyphenation patterns for FOP are made available through the Objects For Formatting Objects project.

> **Note:**
> If you have made improvements to an existing Apache™ FOP hyphenation pattern, or if you have created one from scratch, please consider contributing these to OFFO so that they can benefit other FOP users as well. Please inquire on the FOP User mailing list.

1.2 License Issues

Many of the hyphenation files distributed with TeX and its offspring are licenced under the LaTeX Project Public License (LPPL), which prevents them from being distributed with Apache software. The LPPL puts restrictions on file names in redistributed derived works which we feel can't guarantee. Some hyphenation pattern files have other or additional restrictions, for example against use for commercial purposes.

Although Apache FOP cannot redistribute hyphenation pattern files that do not conform with its license scheme, that does not necessarily prevent users from using such hyphenation patterns with FOP. However, it does place on the user the responsibility for determining whether the user can rightly use such hyphenation patterns under the hyphenation pattern license.

> **Warning:**
> The user is responsible to settle license issues for hyphenation pattern files that are obtained from non-Apache sources.

1.3 Sources of Custom Hyphenation Pattern Files

The most important source of hyphenation pattern files is the CTAN TeX Archive.

1.4 Installing Custom Hyphenation Patterns

To install a custom hyphenation pattern for use with FOP:

1. Convert the TeX hyphenation pattern file to the FOP format. The FOP format is an xml file conforming to the DTD found at `{fop-dir}/hyph/hyphenation.dtd`.
2. Name this new file following this schema: `languageCode_countryCode.xml`. The country code is optional, and should be used only if needed. For example:
   - `en_US.xml` would be the file name for American English hyphenation patterns.
   - `it.xml` would be the file name for Italian hyphenation patterns.

   The language and country codes must match the XSL-FO input, which follows ISO 639 (languages) and ISO 3166 (countries). NOTE: The ISO 639/ISO 3166 convention is that language names are written in lower case, while country codes are written in upper case. FOP does not check whether

the language and country specified in the FO source are actually from the current standard, but it relies on it being two letter strings in a few places. So you can make up your own codes for custom hyphenation patterns, but they should be two letter strings too (patches for proper handling extensions are welcome)

3. There are basically three ways to make the FOP-compatible hyphenation pattern file(s) accessible to FOP:
   - Download the precompiled JAR from OFFO and place it either in the `{fop-dir}/lib` directory, or in a directory of your choice (and append the full path to the JAR to the environment variable `FOP_HYPHENATION_PATH`).
   - Download the desired FOP-compatible hyphenation pattern file(s) from OFFO, and/or take your self created hyphenation pattern file(s),
     - place them in the directory `{fop-dir}/hyph`,
     - or place them in a directory of your choice and set the Ant variable `user.hyph.dir` to point to that directory (in `build-local.properties`),

     and run Ant with build target `jar-hyphenation`. This will create a JAR containing the compiled patterns in `{fop-dir}/build` that will be added to the classpath on the next run. (When FOP is built from scratch, and there are pattern source file(s) present in the directory pointed to by the `user.hyph.dir` variable, this JAR will automatically be created from the supplied pattern(s)).
   - Put the pattern source file(s) into a directory of your choice and configure FOP to look for custom patterns in this directory, by setting the <hyphenation-base> configuration option.

Warning:
Either of these three options will ensure hyphenation is working when using FOP from the command-line. If FOP is being embedded, remember to add the location(s) of the hyphenation JAR(s) to the CLASSPATH (option 1 and 2) or to set the <hyphenation-dir> configuration option programmatically (option 3).

2 Hyphenation Patterns

If you would like to build your own hyphenation pattern files, or modify existing ones, this section will help you understand how to do so. Even when creating a pattern file from scratch, it may be beneficial to start with an existing file and modify it. See OFFO's Hyphenation page for examples. Here is a brief explanation of the contents of FOP's hyphenation patterns:

Warning:
The remaining content of this section should be considered "draft" quality. It was drafted from theoretical literature, and has not been tested against actual FOP behavior. It may contain errors or omissions. Do not rely on these instructions without testing everything stated here. If you use these instructions, please provide feedback on the FOP User mailing list, either confirming their accuracy, or raising specific problems that we can address.

- The root of the pattern file is the <hyphenation-info> element.
- <hyphen-char>: its attribute "value" contains the character signalling a hyphen in the <exceptions> section. It has nothing to do with the hyphenation character used in FOP, use the XSLFO hyphenation-character property for defining the hyphenation character there. At some points a dash U+002D is hardwired in the code, so you'd better use this too (patches to rectify the situation are

welcome). There is no default, if you declare exceptions with hyphenations, you must declare the hyphen-char too.

- <hyphen-min> contains two attributes:
  - before: the minimum number of characters in a word allowed to exist on a line immediately preceding a hyphenated word-break.
  - after: the minimum number of characters in a word allowed to exist on a line immediately after a hyphenated word-break.

  This element is unused and not even read. It should be considered a documentation for parameters used during pattern generation.
- <classes> contains whitespace-separated character sets. The members of each set should be treated as equivalent for purposes of hyphenation, usually upper and lower case of the same character. The first character of the set is the canonical character, the patterns and exceptions should only contain these canonical representation characters (except digits for weight, the period (.) as word delimiter in the patterns and the hyphen char in exceptions, of course).
- <exceptions> contains whitespace-separated words, each of which has either explicit hyphen characters to denote acceptable breakage points, or no hyphen characters, to indicate that this word should never be hyphenated, or contain explicit <hyp> elements for specifying changes of spelling due to hyphenation (like backen -> bak-ken or Stoffarbe -> Stoff-farbe in the old german spelling). Exceptions override the patterns described below. Explicit <hyp> declarations don't work yet (patches welcome). Exceptions are generally a bit brittle, test carefully.
- <patterns> includes whitespace-separated patterns, which are what drive most hyphenation decisions. The characters in these patterns are explained as follows:
  - non-numeric characters represent characters in a sub-word to be evaluated
  - the period character (.) represents a word boundary, i.e. either the beginning or ending of a word
  - numeric characters represent a scoring system for indicating the acceptability of a hyphen in this location. Odd numbers represent an acceptable location for a hyphen, with higher values overriding lower inhibiting values. Even numbers indicate an unacceptable location, with higher values overriding lower values indicating an acceptable position. A value of zero (inhibiting) is implied when there is no number present. Generally patterns are constructed so that valuse greater than 4 are rare. Due to a bug currently patterns with values of 8 and greater don't have an effect, so don't wonder.

  Here are some examples from the English patterns file:
  - Knuth (*The TeXBook*, Appendix H) uses the example **hach4**, which indicates that it is extremely undesirable to place a hyphen after the substring "hach", for example in the word "toothach-es".
  - **.leg5e** indicates that "leg-e", when it occurs at the beginning of a word, is a very good place to place a hyphen, if one is needed. Words like "leg-end" and "leg-er-de-main" fit this pattern.

  Note that the algorithm that uses this data searches for each of the word's substrings in the patterns, and chooses the *highest* value found for letter combination.

If you want to convert a TeX hyphenation pattern file, you have to undo the TeX encoding for non-ASCII text. FOP uses Unicode, and the patterns must be proper Unicode too. You should be aware of the XML encoding issues, preferably use a good Unicode editor.

Note that FOP does not do Unicode character normalization. If you use combining chars for accents and other character decorations, you must declare character classes for them, and use the same sequence of base

character and combining marks in the XSLFO source, otherwise the pattern wouldn't match. Fortunately, Unicode provides precomposed characters for all important cases in common languages, until now nobody run seriously into this issue. Some dead languages and dialects, especially ancient ones, may pose a real problem though.

If you want to generate your own patterns, an open-source utility called patgen can be used to assist in creating pattern files from dictionaries. It is available in many Unix/Linux distributions and every TeX distribution. Pattern creation for languages like english or german is an art. Read Frank Liang's original paper "Word Hy-phen-a-tion by Com-pu-ter" (yes, with hyphens) for details. The original patgen.web source, included in the TeX source distributions, contains valuable comments, unfortunately technical details often obscure the high level issues. Another important source of information is The TeX Book, appendix H (either read the TeX source, or run it through TeX to typeset it). Secondary articles, for example the works by Petr Sojka, may also give some much needed insight into problems arising in automated hyphenation.