The `hyphen.cfg` file for LuaTeX

Khaled Hosny, Élie Roux, and Manuel Pégourié-Gonnard
khaledhosny@eglug.org
elie.roux@telecom-bretagne.eu
mpg@elzevir.fr

2010/04/28 v1.4

Abstract

This is a modified version of the file `hyphen.cfg` distributed with the babel package, with a supporting Lua module, aimed at adapting babel's hyphenation patterns loading mechanism to LuaTeX's dynamic pattern loading capabilities. It makes use of a `language.dat.lua` file (whose format is described below) that should be present in the distribution, in addition to the regular `language.dat` file.

There is a version of `etex.src` modified for the same reasons using similar code, which also makes use of the `luatex-hyphen.lua` and `language.dat.lua` files described here.

1 Documentation

Hyphenation patterns should be loaded at runtime with LuaTeX: if they appear in the format, they will be rehashed when the format is loaded anyway, which makes the format quite long to load (many seconds even on modern machines) and provides for bad user experience. Hence, it is desirable to load as few patterns as possible in the format, and load on-demand the needed patterns at runtime.

This package provides a modified version of hyphen.cfg adapted to LuaTeX, as well as a supporting Lua module. Since a lot of things, especially the catcodes, are not as predictable at runtime than at format creation time, we don't `\input` the usual pattern files, but rather load the patterns using the Lua interface, using a special plain text version of the pattern files if available.

The existence and file name of such a version cannot be guessed, so we need a specific database: the file `language.dat.lua`. This file should be loadable by Lua and return a table whose keys are the canonical language names as found in `language.dat`, and the values are Lua tables consisting of:

1. A fixed part with one mandatory field:

   ```
   synonyms = { <string> alternative name, ...}
   ```

This field's value must be the same as in `language.dat`.

2. A variable part consisting of either:

   - For most languages:

     ```
     patterns = <string> filenames for patterns
     hyphenation = <string> filenames for exceptions
     ```

     Each string contains a coma-separated list of file names (whitespace before or after the coma is not accepted). The files given by `patterns` (resp. `hypenation`) must be plain text files encoded in utf8, with only patterns (resp. exceptions) and not even comments: their content will be used directly without being parsed by TeX. If one of these keys is missing or is the empty string, it is ignored and no patterns (resp. exceptions) are loaded for this language.

   - Special cases are supported by a field `special`. Currently, the following kind of values are recognized:

     `'disabled:<reason>'` allows to disable specific languages: when the user tries to load this language, an error will be issued, with the `<reason>`.

     `'language0'` only `english` should use this type of special, to indicate it is normally dumped in the format as `\language0` (see below).

     Special languages may have `*hyphenmin` information when it makes sense (mostly `\language0`).

3. Optional fields may be added. For example:

   ```
   loader = <string> name of the TeX loader
   lefthyphenmin = <number> value for \lefthyphenmin
   righthyphenmin = <number> value for \righthyphenmin
   ```

   Those fields are present in `language.dat.lua` as generated bu `tlmgr`, for example, but they *are not* used by the present code in any way.

Languages that are mentioned in `language.dat` but not in `language.dat.lua` will be loaded in the format. So, if the `language.dat.lua` file is missing or incomplete, languages will just go back to the "old" behaviour, resulting in longer startup time, which seems less bad than complete breakage.

For backward compatibility, Knuth's original patterns for US English are always loaded in the format, as `\language0`.[1]

The modified version of `hyphen.cfg` provided here checks for the engine, and should continue to work with any engine without any modified behaviour. However, it is recommended to install it in such a way that the original `hyphen.cfg` from babel is found first by any engine other than LuaTeX.

---

[1] It is assumed to be the first entry in `language.dat`.

2 Implementation

2.1 luatex-hyphen.lua

1 ⟨∗lua⟩

Start a Lua module, importing only the necessary functions as locals.

```
2 local error, dofile, pairs, ipairs = error, dofile, pairs, ipairs
3 local io, texio, lang, kpse = io, texio, lang, kpse
4 module('luatexhyphen')
```

Two functions for error and information reporting.

```
5 local function wlog(msg, ...)
6     texio.write_nl('log', 'luatex-hyphen: '..msg:format(...))
7 end
8 local function err(msg, ...)
9     error('luatex-hyphen: '..msg:format(...), 2)
10 end
```

Load the `language.dat.lua` file with the Lua version of the language database.

```
11 local dbname = "language.dat.lua"
12 local language_dat
13 local dbfile = kpse.find_file(dbname)
14 if not dbfile then
15     err("file not found: "..dbname)
16 else
17     wlog('using data file: %s', dbfile)
18     language_dat = dofile(dbfile)
19 end
```

Look up a language in the database, and return the associated information, as well as the canonical name of the language.

```
20 function lookupname(name)
21     if language_dat[name] then
22         return language_dat[name], name
23     else
24         for canon, data in pairs(language_dat) do
25             for _,syn in ipairs(data.synonyms) do
26                 if syn == name then
27                     return data, canon
28                 end
29             end
30         end
31     end
32 end
```

Set hyphenation patterns and exceptions for a language given by its name (in the database) and number (value of `\language`). Doesn't return anything, but will call `error()` if things go wrong.

```
33 function loadlanguage(lname, id)
34     local msg = "loading%s patterns and exceptions for: %s (\\language%d)"
```

Lookup the language in the database.

```
35     local ldata, cname = lookupname(lname)
36     if not ldata then
37         err("no entry in %s for this language: %s", dbname, lname)
38     end
```

Handle special languages.

```
39     if ldata.special then
40         if ldata.special:find('^disabled:') then
41             err("language disabled by %s: %s (%s)", dbname, cname,
42                 ldata.special:gsub('^disabled:', ''))
43         elseif ldata.special == 'language0' then
44             err("\\language0 should be dumped in the format")
45         else
46             err("bad entry in %s for language %s")
47         end
48     end
```

The generic case: load hyphenation patterns and exceptions from files given by the language code.

```
49     wlog(msg, '', cname, id)
50     for _, item in ipairs{'patterns', 'hyphenation'} do
51         local filelist = ldata[item]
52         if filelist ~= nil and filelist ~= '' then
53           for _, file in ipairs(filelist:explode(',')) do
54             local file = kpse.find_file(file) or err("file not found: %s", file)
55             local fh = io.open(file, 'r')
56             local data = fh:read('*a') or err("file not readable: %s", f)
57             fh:close()
58             lang[item](lang.new(id), data)
59           end
60         else
61             if item == 'hyphenation' then item = item..' exceptions' end
62             wlog("info: no %s for this language", item)
63         end
64     end
65 end
```

66 ⟨/lua⟩

2.2   hyphen.cfg

67 ⟨*hyphen⟩

Start with unmodified code from babel.

```
68 \ifx\ProvidesFile\@undefined
69   \def\ProvidesFile#1[#2 #3 #4]{%
70     \wlog{File: #1 #4 #3 <#2>}%
```

Use a modified banner for LuaTEX.

```
71     \ifx\directlua\@undefined
72       \toks8{Babel <#3> and hyphenation patterns for }%
73     \else
74       \toks8{LuaTeX adaptation of babel <#3>
```

4

```
75        and hyphenation patterns for }%
76      \fi
77      \let\ProvidesFile\@undefined
78      }
79   \def\ProvidesLanguage#1[#2 #3 #4]{%
80      \wlog{Language: #1 #4 #3 <#2>}%
81      }
82 \else
83   \let\bbl@tempa\ProvidesFile
84   \def\ProvidesFile#1[#2 #3 #4]{%
```

Same here.

```
85      \ifx\directlua\@undefined
86        \toks8{Babel <#3> and hyphenation patterns for }%
87      \else
88        \toks8{LuaTeX adaptation of babel <#3>
89          and hyphenation patterns for }%
90      \fi
91      \bbl@tempa#1[#2 #3 #4]%
92      \let\ProvidesFile\bbl@tempa}
93   \def\ProvidesLanguage#1{%
94      \begingroup
95        \catcode`\ 10 %
96        \@makeother\/%
97        \@ifnextchar[%
98          {\@provideslanguage{#1}}{\@provideslanguage{#1}[]}}
99   \def\@provideslanguage#1[#2]{%
100     \wlog{Language: #1 #2}%
101     \expandafter\xdef\csname ver@#1.ldf\endcsname{#2}%
102     \endgroup}
103 \fi
104
```

File identification is modified again.

```
105 \ProvidesFile{hyphen.cfg}
106                 [2010/04/26 v3.8l-luatex-1.4 %
107       Language switching mechanism for LuaTeX, adapted from babel v3.8l]
108 \ifx\AtBeginDocument\@undefined
109   \input plain.def\relax
110 \fi
111 \ifx\language\@undefined
112   \csname newcount\endcsname\language
113 \fi
114 \ifx\newlanguage\@undefined
115   \csname newcount\endcsname\last@language
116 \else
117   \countdef\last@language=19
118 \fi
119 \ifx\newlanguage\@undefined
120   \def\addlanguage#1{%
```

```
121    \global\advance\last@language \@ne
122    \ifnum\last@language<\@cclvi
123    \else
124        \errmessage{No room for a new \string\language!}%
125    \fi
126    \global\chardef#1\last@language
127    \wlog{\string#1 = \string\language\the\last@language}}
128 \else
129   \def\addlanguage{\alloc@9\language\chardef\@cclvi}
130 \fi
131 \def\adddialect#1#2{%
132    \global\chardef#1#2\relax
133    \wlog{\string#1 = a dialect from \string\language#2}}
134 \def\iflanguage#1{%
135   \expandafter\ifx\csname l@#1\endcsname\relax
136     \@nolanerr{#1}%
137   \else
138     \bbl@afterfi{\ifnum\csname l@#1\endcsname=\language
139       \expandafter\@firstoftwo
140     \else
141       \expandafter\@secondoftwo
142     \fi}%
143   \fi}
144 \edef\selectlanguage{%
145   \noexpand\protect
146   \expandafter\noexpand\csname selectlanguage \endcsname
147   }
148 \ifx\@undefined\protect\let\protect\relax\fi
149 \ifx\documentclass\@undefined
150   \def\xstring{\string\string\string}
151 \else
152   \let\xstring\string
153 \fi
154 \xdef\bbl@language@stack{}
155 \def\bbl@push@language{%
156   \xdef\bbl@language@stack{\languagename+\bbl@language@stack}%
157   }
158 \def\bbl@pop@lang#1+#2-#3{%
159   \def\languagename{#1}\xdef#3{#2}%
160   }
161 \def\bbl@pop@language{%
162   \expandafter\bbl@pop@lang\bbl@language@stack-\bbl@language@stack
163   \expandafter\bbl@set@language\expandafter{\languagename}%
164   }
165 \expandafter\def\csname selectlanguage \endcsname#1{%
166   \bbl@push@language
167   \aftergroup\bbl@pop@language
168   \bbl@set@language{#1}}
169 \def\bbl@set@language#1{%
170   \edef\languagename{%
```

```
171        \ifnum\escapechar=\expandafter`\string#1\@empty
172        \else \string#1\@empty\fi}%
173    \select@language{\languagename}%
174    \if@filesw
175        \protected@write\@auxout{}{\string\select@language{\languagename}}%
176        \addtocontents{toc}{\xstring\select@language{\languagename}}%
177        \addtocontents{lof}{\xstring\select@language{\languagename}}%
178        \addtocontents{lot}{\xstring\select@language{\languagename}}%
179    \fi}
180 \def\select@language#1{%
181    \expandafter\ifx\csname l@#1\endcsname\relax
182        \@nolanerr{#1}%
183    \else
184        \expandafter\ifx\csname date#1\endcsname\relax
185            \@noopterr{#1}%
186        \else
187            \bbl@patterns{\languagename}%
188            \originalTeX
189            \expandafter\def\expandafter\originalTeX
190                \expandafter{\csname noextras#1\endcsname
191                            \let\originalTeX\@empty}%
192            \languageshorthands{none}%
193            \babel@beginsave
194            \csname captions#1\endcsname
195            \csname date#1\endcsname
196            \csname extras#1\endcsname\relax
197            \babel@savevariable\lefthyphenmin
198            \babel@savevariable\righthyphenmin
199            \expandafter\ifx\csname #1hyphenmins\endcsname\relax
200                \set@hyphenmins\tw@\thr@@\relax
201            \else
202                \expandafter\expandafter\expandafter\set@hyphenmins
203                    \csname #1hyphenmins\endcsname\relax
204            \fi
205        \fi
206    \fi}
207 \long\def\otherlanguage#1{%
208    \csname selectlanguage \endcsname{#1}%
209    \ignorespaces
210    }
211 \long\def\endotherlanguage{%
212    \originalTeX
213    \global\@ignoretrue\ignorespaces
214    }
215 \expandafter\def\csname otherlanguage*\endcsname#1{%
216    \foreign@language{#1}%
217    }
218 \expandafter\def\csname endotherlanguage*\endcsname{%
219    \csname noextras\languagename\endcsname
220    }
```

```
221 \def\foreignlanguage{\protect\csname foreignlanguage \endcsname}
222 \expandafter\def\csname foreignlanguage \endcsname#1#2{%
223   \begingroup
224     \originalTeX
225     \foreign@language{#1}%
226     #2%
227     \csname noextras#1\endcsname
228   \endgroup
229   }
230 \def\foreign@language#1{%
231   \def\languagename{#1}%
232   \expandafter\ifx\csname l@#1\endcsname\relax
233     \@nolanerr{#1}%
234   \else
235     \bbl@patterns{\languagename}%
236     \languageshorthands{none}%
237     \csname extras#1\endcsname
238     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
239       \set@hyphenmins\tw@\thr@@\relax
240     \else
241       \expandafter\expandafter\expandafter\set@hyphenmins
242         \csname #1hyphenmins\endcsname\relax
243     \fi
244   \fi
245   }
246 \def\bbl@patterns#1{%
247   \language=\expandafter\ifx\csname l@#1:\f@encoding\endcsname\relax
248     \csname l@#1\endcsname
249   \else
250     \csname l@#1:\f@encoding\endcsname
251   \fi\relax
```

With LuaTeX, load patterns and exceptions on the fly using functions from the supporting Lua module, unless of course they are already loaded for this language (identified by its number to avoid problems with synonyms).

Also, since this code will be executed at runtime, be careful while testing if we're using LuaTeX.

```
252   \ifx\directlua\@undefined\else
253     \ifx\directlua\relax\else
254       \ifcsname lu@texhyphen@loaded@\the\language\endcsname \else
255         \global\@namedef{lu@texhyphen@loaded@\the\language}{}%
256         \directlua{
257           if not luatexhyphen then
258               dofile(assert(kpse.find_file("luatex-hyphen.lua")))
259           end
260           luatexhyphen.loadlanguage("\luatexluaescapestring{#1}",
261             \the\language)}%
262       \fi
263     \fi
264   \fi
```

```
265 }
266 \def\hyphenrules#1{%
267   \expandafter\ifx\csname l@#1\endcsname\@undefined
268     \@nolanerr{#1}%
269   \else
270     \bbl@patterns{#1}%
271     \languageshorthands{none}%
272       \expandafter\ifx\csname #1hyphenmins\endcsname\relax
273         \set@hyphenmins\tw@\thr@@\relax
274       \else
275         \expandafter\expandafter\expandafter\set@hyphenmins
276         \csname #1hyphenmins\endcsname\relax
277       \fi
278   \fi
279   }
280 \def\endhyphenrules{}
281 \def\providehyphenmins#1#2{%
282   \expandafter\ifx\csname #1hyphenmins\endcsname\relax
283     \@namedef{#1hyphenmins}{#2}%
284   \fi}
285 \def\set@hyphenmins#1#2{\lefthyphenmin#1\righthyphenmin#2}
286 \def\LdfInit{%
287   \chardef\atcatcode=\catcode`\@
288   \catcode`\@=11\relax
289   \input babel.def\relax
290   \catcode`\@=\atcatcode \let\atcatcode\relax
291   \LdfInit}
292 \ifx\originalTeX\@undefined\let\originalTeX\@empty\fi
293 \ifx\babel@beginsave\@undefined\let\babel@beginsave\relax\fi
294 \ifx\PackageError\@undefined
295   \def\@nolanerr#1{%
296     \errhelp{Your command will be ignored, type <return> to proceed}%
297     \errmessage{You haven't defined the language #1\space yet}}
298   \def\@nopatterns#1{%
299     \message{No hyphenation patterns were loaded for}%
300     \message{the language `#1'}%
301     \message{I will use the patterns loaded for \string\language=0
302           instead}}
303   \def\@noopterr#1{%
304     \errmessage{The option #1 was not specified in \string\usepackage}
305     \errhelp{You may continue, but expect unexpected results}}
306   \def\@activated#1{%
307     \wlog{Package babel Info: Making #1 an active character}}
308 \else
309   \newcommand*{\@nolanerr}[1]{%
310     \PackageError{babel}%
311               {You haven't defined the language #1\space yet}%
312         {Your command will be ignored, type <return> to proceed}}
313   \newcommand*{\@nopatterns}[1]{%
314     \PackageWarningNoLine{babel}%
```

```
315        {No hyphenation patterns were loaded for\MessageBreak
316           the language `#1'\MessageBreak
317           I will use the patterns loaded for \string\language=0
318           instead}}
319  \newcommand*{\@noopterr}[1]{%
320     \PackageError{babel}%
321                     {You haven't loaded the option #1\space yet}%
322                 {You may proceed, but expect unexpected results}}
323  \newcommand*{\@activated}[1]{%
324     \PackageInfo{babel}{%
325        Making #1 an active character}}
326  \fi
327  \def\process@line#1#2 #3/{%
328    \ifx=#1
329      \process@synonym#2 /
330    \else
331      \process@language#1#2 #3/%
332    \fi
333    }
334  \toks@{}
335  \def\process@synonym#1 /{%
336    \ifnum\last@language=\m@ne
337      \expandafter\chardef\csname l@#1\endcsname0\relax
338      \wlog{\string\l@#1=\string\language0}
339      \toks@\expandafter{\the\toks@
340        \expandafter\let\csname #1hyphenmins\expandafter\endcsname
341        \csname\languagename hyphenmins\endcsname}%
342    \else
343      \expandafter\chardef\csname l@#1\endcsname\last@language
344      \wlog{\string\l@#1=\string\language\the\last@language}
345      \expandafter\let\csname #1hyphenmins\expandafter\endcsname
346      \csname\languagename hyphenmins\endcsname
347    \fi
348    }
349  \def\process@language#1 #2 #3/{%
350    \expandafter\addlanguage\csname l@#1\endcsname
351    \expandafter\language\csname l@#1\endcsname
352    \def\languagename{#1}%
```

In the LuaTEXcase, we have to decide wether to load the language now. Remember our choice, since we'll need it two times more.

If we choose to load the language now, mark it as loaded. This is done using TEX macros in order to survive the format dumping-loading cycle, which would not be as straigthforward using Lua objects.

```
353    \ifx\directlua\@undefined
354      \global\toks8\expandafter{\the\toks8#1, }%
355    \else
356      \directlua{
357        if not luatexhyphen then
358          dofile(assert(kpse.find_file("luatex-hyphen.lua")))
```

```
359        end
360        processnow = (tex.language == 0) or
361          (luatexhyphen.lookupname("\luatexluaescapestring{#1}") == nil)}%
362      \ifnum0=\directlua{tex.sprint(processnow and "0" or "1")}\relax
363        \global\toks8\expandafter{\the\toks8#1, }%
364        \global\@namedef{lu@texhyphen@loaded@\the\language}{}%
365      \fi
366    \fi
367    \begingroup
368      \bbl@get@enc#1:\@@@
369      \ifx\bbl@hyph@enc\@empty
370      \else
371        \fontencoding{\bbl@hyph@enc}\selectfont
372      \fi
373      \lefthyphenmin\m@ne
```

Conditionally input the patterns file.

```
374      \ifx\directlua\@undefined
375        \input #2\relax
376      \else
377        \ifnum0=\directlua{tex.sprint(processnow and "0" or "1")}\relax
378          \input #2\relax
379        \fi
380      \fi
381      \ifnum\lefthyphenmin=\m@ne
382      \else
383        \expandafter\xdef\csname #1hyphenmins\endcsname{%
384          \the\lefthyphenmin\the\righthyphenmin}%
385      \fi
386    \endgroup
387    \ifnum\the\language=\z@
388      \expandafter\ifx\csname #1hyphenmins\endcsname\relax
389        \set@hyphenmins\tw@\thr@@\relax
390      \else
391        \expandafter\expandafter\expandafter\set@hyphenmins
392          \csname #1hyphenmins\endcsname
393      \fi
394      \the\toks@
395    \fi
396    \toks@{}%
397    \def\bbl@tempa{#3}%
398    \ifx\bbl@tempa\@empty
399    \else
400      \ifx\bbl@tempa\space
401      \else
```

Conditionnaly input the exceptions file.

```
402        \ifx\directlua\@undefined
403          \input #3\relax
404        \else
```

```
405      \ifnum0=\directlua{tex.sprint(processnow and "0" or "1")}\relax
406        \input #3\relax
407      \fi
408      \directlua{processnow = nil}%
409    \fi
410    \fi
411  \fi
412  }
413 \def\bbl@get@enc#1:#2\@@@{%
414   \def\bbl@tempa{#1}%
415   \def\bbl@tempb{#2}%
416   \ifx\bbl@tempb\@empty
417     \let\bbl@hyph@enc\@empty
418   \else
419     \bbl@get@enc#2\@@@
420     \edef\bbl@hyph@enc{\bbl@tempa}%
421   \fi}
422 \openin1 = language.dat
423 \ifeof1
424   \message{I couldn't find the file language.dat,\space
425           I will try the file hyphen.tex}
426   \input hyphen.tex\relax
427 \else
428   \last@language\m@ne
429   \loop
430     \endlinechar\m@ne
431     \read1 to \bbl@line
432     \endlinechar`\^^M
433     \ifx\bbl@line\@empty
434     \else
435       \edef\bbl@line{\bbl@line\space/}%
436       \expandafter\process@line\bbl@line
437     \fi
438     \iftrue \csname fi\endcsname
439     \csname if\ifeof1 false\else true\fi\endcsname
440   \repeat
441   \language=0
442 \fi
443 \closein1
444 \let\process@language\@undefined
445 \let\process@synonym\@undefined
446 \let\process@line\@undefined
447 \let\bbl@tempa\@undefined
448 \let\bbl@tempb\@undefined
449 \let\bbl@eq@\@undefined
450 \let\bbl@line\@undefined
451 \let\bbl@get@enc\@undefined
452 \ifx\addto@hook\@undefined
453 \else
```

```
454    \expandafter\addto@hook\expandafter\everyjob\expandafter{%
455      \expandafter\typeout\expandafter{\the\toks8 loaded.}}
456 \fi

457 ⟨/hyphen⟩
```