

The `hyphen.cfg` file for Lua_T_EX

Khaled Hosny, Élie Roux, and Manuel Pégourié-Gonnard

`khaledhosny@eglug.org`

`elie.roux@telecom-bretagne.eu`

`mpg@elzevir.fr`

2010/04/28 v1.3beta

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 Lua_T_EX's dynamic pattern loading capabilities.

1 Documentation

Hyphenation patterns should be loaded at runtime with Lua_T_EX: 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.

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

This package provides a modified version of `hyphen.cfg` adapted to Lua_T_EX, 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 kindly provided by the `texhyphen` project.

The modifications applied in this files are highlighted in the code documentation below, but here is a summary:

- not loading patterns in the format except for english
- loading patterns at runtime, except for english
- modified banner

This file 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`.

2 Package code

2.1 `luatex-hyphen.lua`

```
1
2 luatexhyphen = {}
3
4 luatexhyphen.version = "1.3beta"
5
6 local dbname = "language.dat.lua"
7
8 local function warn (msg, ...)
9     texio.write_nl('luatex-hyphen: '..string.format(msg, ...))
10 end
11
12 luatexhyphen.language_dat = {}
13 local dbfile = kpse.find_file(dbname)
14 if not dbfile then
15     warn("file not found: "..dbname)
16 else
17     luatexhyphen.language_dat = dofile(dbfile)
18 end
19
20 local function lookupname(l)
21     if luatexhyphen.language_dat[l] then
22         return luatexhyphen.language_dat[l], 1
23     else
24         for orig,lt in pairs(luatexhyphen.language_dat) do
25             for _,syn in ipairs(lt.synonyms) do
26                 if syn == l then
27                     return lt, orig
28                 end
29             end
30         end
31     end
32     return nil
33 end
34
35 function luatexhyphen.loadpatterns(l, id)
36     local lt, orig = lookupname(l)
37     if not lt or not lt.code then
38         warn("no entry in %s for this language: %s", dbname, l)
39         return
40     end
41     local n = 'hyph-'..lt.code..'pat.txt'
42     local f = kpse.find_file(n)
```

```

43     if not f then
44         warn("file not found: %s", n)
45         return
46     end
47     f = io.open(f, 'r')
48     local data = f:read('*a')
49     f:close()
50     if not data then
51         warn("file not readable: %s", f)
52         return
53     end
54     local lobj = lang.new(id)
55     warn("loading patterns for: %s", orig)
56     lang.patterns(lobj, data)
57 end
58
59 function luatexhyphen.loadexceptions(l, id)
60     local lt, orig = lookupname(l)
61     if not lt or not lt.code then
62         warn("no entry in %s for this language: %s", dbname, l)
63         return
64     end
65     local n = 'hyph-'..lt.code..'hyp.txt'
66     local f = kpse.find_file(n)
67     if not f then
68         warn("file not found: %s", n)
69         return
70     end
71     f = io.open(f, 'r')
72     local data = f:read('*a')
73     f:close()
74     if not data then
75         warn("file not readable: %s", f)
76         return
77     end
78     local lobj = lang.new(id)
79     warn("loading exceptions for: %s", orig)
80     lang.hyphenation(lobj, data)
81 end
82

```

2.2 hyphen.cfg

```

83 \ifx\ProvidesFile\@undefined
84 \def\ProvidesFile#1[#2 #3 #4]{%
85     \wlog{File: #1 #4 #3 <#2>}%

    Use a modified banner for LuaTeX.
86 \ifx\directlua\@undefined
87     \toks8{Babel <#3> and hyphenation patterns for }%
88 \else

```

```

89     \toks8{LuaTeX adaptation of babel <#3>
90     and hyphenation patterns for }%
91     \fi

92     \let\ProvidesFile\@undefined
93     }
94     \def\ProvidesLanguage#1[#2 #3 #4]{%
95         \wlog{Language: #1 #4 #3 <#2>}%
96     }
97 \else
98     \let\bbl@tempa\ProvidesFile
99     \def\ProvidesFile#1[#2 #3 #4]{%

Same here.
100     \ifx\directlua\@undefined
101         \toks8{Babel <#3> and hyphenation patterns for }%
102     \else
103         \toks8{LuaTeX adaptation of babel <#3>
104         and hyphenation patterns for }%
105     \fi

106     \bbl@tempa#1[#2 #3 #4]%
107     \let\ProvidesFile\bbl@tempa}
108     \def\ProvidesLanguage#1{%
109         \begingroup
110         \catcode'\ 10 %
111         \@makeother\/%
112         \@ifnextchar[%]
113             {\@provideslanguage{#1}}{\@provideslanguage{#1}[]}
114     \def\@provideslanguage#1[#2]{%
115         \wlog{Language: #1 #2}%
116         \expandafter\xdef\csname ver@#1.1df\endcsname{#2}%
117     \endgroup}
118 \fi
119

```

File identification is modified again.

```

120 \ProvidesFile{hyphen.cfg}
121     [2010/04/26 v3.81-luatex-1.3beta %
122     Language switching mechanism for LuaTeX, adapted from babel v3.81]

123 \ifx\AtBeginDocument\@undefined
124     \input plain.def\relax
125 \fi

126 \ifx\language\@undefined
127     \csname newcount\endcsname\language
128 \fi

129 \ifx\newlanguage\@undefined
130     \csname newcount\endcsname\last@language
131 \else
132     \countdef\last@language=19
133 \fi

134 \ifx\newlanguage\@undefined

```

```

135 \def\addlanguage#1{%
136   \global\advance\last@language \@ne
137   \ifnum\last@language<\@cclvi
138   \else
139     \errmessage{No room for a new \string\language!}%
140   \fi
141   \global\chardef#1\last@language
142   \wlog{\string#1 = \string\language\the\last@language}}
143 \else
144   \def\addlanguage{\alloc@9\language\chardef\@cclvi}
145 \fi
146 \def\adddialect#1#2{%
147   \global\chardef#1#2\relax
148   \wlog{\string#1 = a dialect from \string\language#2}}
149 \def\iflanguage#1{%
150   \expandafter\ifx\csname l@#1\endcsname\relax
151   \@nolanerr{#1}%
152   \else
153     \bbl@afterfi{\ifnum\csname l@#1\endcsname=\language
154       \expandafter\@firstoftwo
155     \else
156       \expandafter\@secondoftwo
157     \fi}%
158   \fi}
159 \edef\selectlanguage{%
160   \noexpand\protect
161   \expandafter\noexpand\csname selectlanguage \endcsname
162 }
163 \ifx\@undefined\protect\let\protect\relax\fi
164 \ifx\documentclass\@undefined
165   \def\xstring{\string\string\string}
166 \else
167   \let\xstring\string
168 \fi
169 \xdef\bbl@language@stack{}
170 \def\bbl@push@language{%
171   \xdef\bbl@language@stack{\language+\bbl@language@stack}%
172 }
173 \def\bbl@pop@lang#1+#2-#3{%
174   \def\language#1\def#3{#2}%
175 }
176 \def\bbl@pop@language{%
177   \expandafter\bbl@pop@lang\bbl@language@stack-\bbl@language@stack
178   \expandafter\bbl@set@language\expandafter{\language}%
179 }
180 \expandafter\def\csname selectlanguage \endcsname#1{%
181   \bbl@push@language
182   \aftergroup\bbl@pop@language
183   \bbl@set@language{#1}}
184 \def\bbl@set@language#1{%

```

```

185 \edef\language{\language}%
186 \ifnum\escapechar=\expandafter'\string#1\@empty
187 \else \string#1\@empty\fi}%
188 \select@language{\language}%
189 \if@files
190 \protected@write\@auxout{}\{\string\select@language{\language}\}%
191 \addtocontents{toc}\{\xstring\select@language{\language}\}%
192 \addtocontents{lof}\{\xstring\select@language{\language}\}%
193 \addtocontents{lot}\{\xstring\select@language{\language}\}%
194 \fi}
195 \def\select@language#1{%
196 \expandafter\ifx\csname l@#1\endcsname\relax
197 \@nolanerr{#1}%
198 \else
199 \expandafter\ifx\csname date#1\endcsname\relax
200 \@noopterr{#1}%
201 \else
202 \bbl@patterns{\language}%
203 \originalTeX
204 \expandafter\def\expandafter\originalTeX
205 \expandafter{\csname noextras#1\endcsname
206 \let\originalTeX\@empty}%
207 \languageshorthands{none}%
208 \babel@beginsave
209 \csname captions#1\endcsname
210 \csname date#1\endcsname
211 \csname extras#1\endcsname\relax
212 \babel@savevariable\lefthyphenmin
213 \babel@savevariable\righthyphenmin
214 \expandafter\ifx\csname #1hyphenmins\endcsname\relax
215 \set@hyphenmins\tw@\thr@@\relax
216 \else
217 \expandafter\expandafter\expandafter\set@hyphenmins
218 \csname #1hyphenmins\endcsname\relax
219 \fi
220 \fi
221 \fi}
222 \long\def\otherlanguage#1{%
223 \csname selectlanguage \endcsname{#1}%
224 \ignorespaces
225 }
226 \long\def\endotherlanguage{%
227 \originalTeX
228 \global\@ignoretrue\ignorespaces
229 }
230 \expandafter\def\csname otherlanguage*\endcsname#1{%
231 \foreign@language{#1}%
232 }
233 \expandafter\def\csname endotherlanguage*\endcsname{%
234 \csname noextras\language\endcsname

```

```

235 }
236 \def\foreignlanguage{\protect\csname foreignlanguage \endcsname}
237 \expandafter\def\csname foreignlanguage \endcsname#1#2{%
238   \begingroup
239     \originalTeX
240     \foreign@language{#1}%
241     #2%
242     \csname noextras#1\endcsname
243   \endgroup
244 }
245 \def\foreign@language#1{%
246   \def\languagename{#1}%
247   \expandafter\ifx\csname l@#1\endcsname\relax
248     \nolannerr{#1}%
249   \else
250     \bbl@patterns{\languagename}%
251     \languageshorthands{none}%
252     \csname extras#1\endcsname
253     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
254       \set@hyphenmins\tw@\thr@@\relax
255     \else
256       \expandafter\expandafter\expandafter\set@hyphenmins
257       \csname #1hyphenmins\endcsname\relax
258     \fi
259   \fi
260 }
261 \def\bbl@patterns#1{%

```

With Lua_T_EX, load patterns and exceptions at runtime using functions from the supporting Lua module.

Remember which patterns have been loaded to avoid reloading patterns and exceptions every time the language is activated. This is done in _T_EX rather than in Lua so that the information about which patterns are loaded in the format is correctly remembered at runtime.¹

```

262 \ifx\directlua@\undefined\else
263   \unless\ifcsname bbl@luatex@#1@loaded\endcsname
264     \expandafter\gdef\csname bbl@luatex@#1@loaded\endcsname{%
265       \directlua{
266         if not luatexhyphen then
267           dofile(kpse.find_file("luatex-hyphen.lua"))
268         end
269         luatexhyphen.loadpatterns("\luatexluaescapestring{#1}",
270           \number\csname l@#1\endcsname)
271         luatexhyphen.loadexceptions("\luatexluaescapestring{#1}",
272           \number\csname l@#1\endcsname)
273       }%

```

¹It is theoretically possible to do so in Lua too, saving things in a bytecode register and restoring if via `\everyjob`, but there is currently no standard mechanism for this, and the _T_EX works well anyway.

```

274 \fi
275 \fi

276 \language=\expandafter\ifx\csname l@#1:\f@encoding\endcsname\relax
277 \csname l@#1\endcsname
278 \else
279 \csname l@#1:\f@encoding\endcsname
280 \fi\relax
281 }

282 \def\hyphenrules#1{%
283 \expandafter\ifx\csname l@#1\endcsname\@undefined
284 \nolanerr{#1}%
285 \else
286 \bbl@patterns{#1}%
287 \languageshorthands{none}%
288 \expandafter\ifx\csname #1hyphenmins\endcsname\relax
289 \set@hyphenmins\tw@\thr@@\relax
290 \else
291 \expandafter\expandafter\expandafter\set@hyphenmins
292 \csname #1hyphenmins\endcsname\relax
293 \fi
294 \fi
295 }

296 \def\endhyphenrules{}
297 \def\providehyphenmins#1#2{%
298 \expandafter\ifx\csname #1hyphenmins\endcsname\relax
299 \namedef{#1hyphenmins}{#2}%
300 \fi}
301 \def\set@hyphenmins#1#2{\lefthyphenmin#1\righthyphenmin#2}
302 \def\LdfInit{%
303 \chardef\atcatcode=\catcode'\@
304 \catcode'\@=11\relax
305 \input babel.def\relax
306 \catcode'\@=\atcatcode \let\atcatcode\relax
307 \LdfInit}
308 \ifx\originalTeX\@undefined\let\originalTeX\@empty\fi
309 \ifx\babel@beginsave\@undefined\let\babel@beginsave\relax\fi
310 \ifx\PackageError\@undefined
311 \def\@nolanerr#1{%
312 \errhelp{Your command will be ignored, type <return> to proceed}%
313 \errmessage{You haven't defined the language #1\space yet}}
314 \def\@nopatterns#1{%
315 \message{No hyphenation patterns were loaded for}%
316 \message{the language '#1'}%
317 \message{I will use the patterns loaded for \string\language=0
318 instead}}
319 \def\@noopterr#1{%
320 \errmessage{The option #1 was not specified in \string\usepackage}
321 \errhelp{You may continue, but expect unexpected results}}
322 \def\@activated#1{%

```



```

323     \wlog{Package babel Info: Making #1 an active character}}
324 \else
325   \newcommand*{\@nolanerr}[1]{%
326     \PackageError{babel}%
327       {You haven't defined the language #1\space yet}%
328       {Your command will be ignored, type <return> to proceed}}
329   \newcommand*{\@nopatterns}[1]{%
330     \PackageWarningNoLine{babel}%
331       {No hyphenation patterns were loaded for\MessageBreak
332         the language '#1'\MessageBreak
333         I will use the patterns loaded for \string\language=0
334         instead}}
335   \newcommand*{\@noopterr}[1]{%
336     \PackageError{babel}%
337       {You haven't loaded the option #1\space yet}%
338       {You may proceed, but expect unexpected results}}
339   \newcommand*{\@activated}[1]{%
340     \PackageInfo{babel}{%
341       Making #1 an active character}}
342 \fi
343 \def\process@line#1#2 #3/{%
344   \ifx=#1
345     \process@synonym#2 /
346   \else
347     \process@language#1#2 #3/%
348   \fi
349 }
350 \toks@{}
351 \def\process@synonym#1 /{%
352   \ifnum\last@language=\m@ne
353     \expandafter\chardef\csname l@#1\endcsname\relax
354     \wlog{\string\l@#1=\string\language0}
355     \toks@\expandafter{\the\toks@
356       \expandafter\let\csname #1hyphenmins\expandafter\endcsname
357       \csname\language\hyphenmins\endcsname}%
358   \else
359     \expandafter\chardef\csname l@#1\endcsname\last@language
360     \wlog{\string\l@#1=\string\language\the\last@language}
361     \expandafter\let\csname #1hyphenmins\expandafter\endcsname
362     \csname\language\hyphenmins\endcsname
363   \fi
364 }
365 \def\process@language#1 #2 #3/{%
366   \expandafter\addlanguage\csname l@#1\endcsname
367   \expandafter\language\csname l@#1\endcsname
368   \def\language{#1}%

```

Yet another banner modification. See below why the test makes sense.

```

369   \ifx\directlua\@undefined
370     \global\toks8\expandafter{\the\toks8#1, }%

```

```

371 \else
372   \unless\ifcsname bbl@luatex@english@loaded\endcsname
373     \global\toks8\expandafter{\the\toks8#1 }%
374   \fi
375 \fi

376 \begingroup
377   \bbl@get@enc#1:\@@@
378   \ifx\bbl@hyph@enc\@empty
379   \else
380     \fontencoding{\bbl@hyph@enc}\selectfont
381   \fi
382   \lefthyphenmin\m@ne

```

Assume the first (that is, zeroth) language in `language.dat` is English. This assumption is very reasonable, since otherwise it would break compatibility with frozen `TeX`by not providing Knuth's original patterns as `\language0`, so we're pretty sure about this point.

We do load this first language, since we want Knuth's patterns to be active as soon as the format is loaded. But once it is done, we don't want to load any other language.

```

383 \ifx\directlua\@undefined
384   \input #2\relax
385 \else
386   \unless\ifcsname bbl@luatex@english@loaded\endcsname
387     \gdef\bbl@luatex@english@loaded{1}%
388     \input #2\relax
389   \fi
390 \fi

391 \ifnum\lefthyphenmin=\m@ne
392 \else
393   \expandafter\xdef\csname #1hyphenmins\endcsname{%
394     \the\lefthyphenmin\the\righthyphenmin}%
395   \fi
396 \endgroup
397 \ifnum\the\language=\z@
398   \expandafter\ifx\csname #1hyphenmins\endcsname\relax
399     \set@hyphenmins\tw@\thr@@\relax
400   \else
401     \expandafter\expandafter\expandafter\set@hyphenmins
402       \csname #1hyphenmins\endcsname
403   \fi
404   \the\toks@
405 \fi
406 \toks@{}%
407 \def\bbl@tempa{#3}%
408 \ifx\bbl@tempa\@empty
409 \else
410   \ifx\bbl@tempa\space
411   \else

```

Likewise, don't load hyphenation exceptions now, but rather when we load the patterns. (Anyway, in practice, the third field of `language.dat` is never used since exceptions are defined in the same file as patterns, so it doesn't really matter.)

There are no hyphenation exceptions for english, and since it is frozen, we can rely on this, so no need for a special case for english here.

```

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

```

```
456 \let\bbl@get@enc\@undefined
457 \ifx\addto@hook\@undefined
458 \else
459   \expandafter\addto@hook\expandafter\everyjob\expandafter{%
460     \expandafter\typeout\expandafter{\the\toks8 loaded.}}
461 \fi
```