

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.loadlanguage(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     warn("loading patterns and exceptions for: %s (\\language%s)", orig, id)
42     for _, ext in ipairs({'pat', 'hyp'}) do
```

```

43     local n = 'hyph-'.lt.code..'..ext..'.txt'
44     local f = kpse.find_file(n)
45     if not f then
46         warn("file not found: %s", n)
47         return
48     end
49     f = io.open(f, 'r')
50     local data = f:read('*a')
51     f:close()
52     if not data then
53         warn("file not readable: %s", f)
54         return
55     end
56     local lobj = lang.new(id)
57     lang.patterns(lobj, data)
58 end
59 end

```

2.2 hyphen.cfg

```

60 \ifx\ProvidesFile\@undefined
61   \def\ProvidesFile#1[#2 #3 #4]{%
62     \wlog{File: #1 #4 #3 <#2>}%

    Use a modified banner for LuaTeX.
63   \ifx\directlua\@undefined
64     \toks8{Babel <#3> and hyphenation patterns for }%
65   \else
66     \toks8{LuaTeX adaptation of babel <#3>
67       and hyphenation patterns for }%
68   \fi
69   \let\ProvidesFile\@undefined
70 }
71 \def\ProvidesLanguage#1[#2 #3 #4]{%
72   \wlog{Language: #1 #4 #3 <#2>}%
73 }
74 \else
75   \let\bbl@tempa\ProvidesFile
76   \def\ProvidesFile#1[#2 #3 #4]{%

    Same here.
77   \ifx\directlua\@undefined
78     \toks8{Babel <#3> and hyphenation patterns for }%
79   \else
80     \toks8{LuaTeX adaptation of babel <#3>
81       and hyphenation patterns for }%
82   \fi
83   \bbl@tempa#1[#2 #3 #4]%
84   \let\ProvidesFile\bbl@tempa}
85 \def\ProvidesLanguage#1{%
86   \begingroup

```

```

87     \catcode'\ 10 %
88     \@makeother\/%
89     \@ifnextchar[%]
90         {\@provideslanguage{#1}}{\@provideslanguage{#1}[]}}
91 \def\@provideslanguage#1[#2]{%
92     \wlog{Language: #1 #2}%
93     \expandafter\xdef\csname ver@#1.ldf\endcsname{#2}%
94     \endgroup}
95 \fi
96
    File identification is modified again.
97 \ProvidesFile{hyphen.cfg}
98     [2010/04/26 v3.81-luatex-1.3beta %
99     Language switching mechanism for LuaTeX, adapted from babel v3.81]
100 \ifx\AtBeginDocument\@undefined
101     \input plain.def\relax
102 \fi
103 \ifx\language\@undefined
104     \csname newcount\endcsname\language
105 \fi
106 \ifx\newlanguage\@undefined
107     \csname newcount\endcsname\last@language
108 \else
109     \countdef\last@language=19
110 \fi
111 \ifx\newlanguage\@undefined
112     \def\addlanguage#1{%
113         \global\advance\last@language \@ne
114         \ifnum\last@language<\ccclvi
115             \else
116                 \errmessage{No room for a new \string\language!}%
117             \fi
118             \global\chardef#1\last@language
119             \wlog{\string#1 = \string\language\the\last@language}}
120 \else
121     \def\addlanguage{\alloc@9\language\chardef\@ccclvi}
122 \fi
123 \def\adddialect#1#2{%
124     \global\chardef#1#2\relax
125     \wlog{\string#1 = a dialect from \string\language#2}}
126 \def\iflanguage#1{%
127     \expandafter\ifx\csname l@#1\endcsname\relax
128         \@nolanerr{#1}%
129     \else
130         \bbl@afterfi{\ifnum\csname l@#1\endcsname=\language
131             \expandafter\@firstoftwo
132         \else
133             \expandafter\@secondoftwo
134         \fi}%

```

```

135 \fi}
136 \edef\selectlanguage{%
137 \noexpand\protect
138 \expandafter\noexpand\csname selectlanguage \endcsname
139 }
140 \ifx\@undefined\protect\let\protect\relax\fi
141 \ifx\documentclass\@undefined
142 \def\xstring{\string\string\string}
143 \else
144 \let\xstring\string
145 \fi
146 \xdef\bbl@language@stack{}
147 \def\bbl@push@language{%
148 \xdef\bbl@language@stack{\language+\bbl@language@stack}%
149 }
150 \def\bbl@pop@lang#1+#2-#3{%
151 \def\language#1}\xdef#3{#2}%
152 }
153 \def\bbl@pop@language{%
154 \expandafter\bbl@pop@lang\bbl@language@stack-\bbl@language@stack
155 \expandafter\bbl@set@language\expandafter{\language}%
156 }
157 \expandafter\def\csname selectlanguage \endcsname#1{%
158 \bbl@push@language
159 \aftergroup\bbl@pop@language
160 \bbl@set@language{#1}}
161 \def\bbl@set@language#1{%
162 \edef\language{%
163 \ifnum\escapechar=\expandafter'\string#1\@empty
164 \else \string#1\@empty\fi}%
165 \select@language{\language}%
166 \if@filesw
167 \protected@write\@auxout{\string\select@language{\language}}%
168 \addtocontents{toc}{\xstring\select@language{\language}}%
169 \addtocontents{lof}{\xstring\select@language{\language}}%
170 \addtocontents{lot}{\xstring\select@language{\language}}%
171 \fi}
172 \def\select@language#1{%
173 \expandafter\ifx\csname l@#1\endcsname\relax
174 \@nolanerr{#1}%
175 \else
176 \expandafter\ifx\csname date#1\endcsname\relax
177 \@noopterr{#1}%
178 \else
179 \bbl@patterns{\language}%
180 \originalTeX
181 \expandafter\def\expandafter\originalTeX
182 \expandafter{\csname noextras#1\endcsname
183 \let\originalTeX\@empty}%
184 \languageshorthands{none}%

```

```

185     \babel@beginsave
186     \csname captions#1\endcsname
187     \csname date#1\endcsname
188     \csname extras#1\endcsname\relax
189     \babel@savevariable\lefthyphenmin
190     \babel@savevariable\righthyphenmin
191     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
192     \set@hyphenmins\tw@\thr@@\relax
193     \else
194     \expandafter\expandafter\expandafter\set@hyphenmins
195     \csname #1hyphenmins\endcsname\relax
196     \fi
197   \fi
198 \fi}
199 \long\def\otherlanguage#1{%
200   \csname selectlanguage \endcsname{#1}%
201   \ignorespaces
202 }
203 \long\def\endotherlanguage{%
204   \originalTeX
205   \global\@ignoretrue\ignorespaces
206 }
207 \expandafter\def\csname otherlanguage*\endcsname#1{%
208   \foreign@language{#1}%
209 }
210 \expandafter\def\csname endotherlanguage*\endcsname{%
211   \csname noextras\language\endcsname
212 }
213 \def\foreignlanguage{\protect\csname foreignlanguage \endcsname}
214 \expandafter\def\csname foreignlanguage \endcsname#1#2{%
215   \begingroup
216     \originalTeX
217     \foreign@language{#1}%
218     #2%
219     \csname noextras#1\endcsname
220   \endgroup
221 }
222 \def\foreign@language#1{%
223   \def\language#1}%
224   \expandafter\ifx\csname l@#1\endcsname\relax
225     \@nolanerr{#1}%
226   \else
227     \bbl@patterns{\language}%
228     \languageshortands{none}%
229     \csname extras#1\endcsname
230     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
231     \set@hyphenmins\tw@\thr@@\relax
232   \else
233     \expandafter\expandafter\expandafter\set@hyphenmins
234     \csname #1hyphenmins\endcsname\relax

```

```

235   \fi
236   \fi
237 }
238 \def\bbl@patterns#1{%

    With LuaTeX, 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 TeX rather than
    in Lua so that the information about which patterns are loaded in the format is
    correctly remembered at runtime.1
239   \ifx\directlua@\undefined\else
240     \unless\ifcsname bbl@luatex@#1@loaded\endcsname
241       \expandafter\gdef\csname bbl@luatex@#1@loaded\endcsname{%
242         \directlua{
243           if not luatexhyphen then
244             dofile(kpse.find_file("luatex-hyphen.lua"))
245           end
246           luatexhyphen.loadlanguage("\luatexluaescapestring{#1}",
247             \number\csname l@#1\endcsname)
248         }%
249       \fi
250     \fi

251   \language=\expandafter\ifx\csname l@#1:\f@encoding\endcsname\relax
252     \csname l@#1\endcsname
253   \else
254     \csname l@#1:\f@encoding\endcsname
255   \fi\relax
256 }

257 \def\hyphenrules#1{%
258   \expandafter\ifx\csname l@#1\endcsname@\undefined
259     \@nolanerr{#1}%
260   \else
261     \bbl@patterns{#1}%
262     \languageshorthands{none}%
263     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
264       \set@hyphenmins\tw@\thr@@\relax
265     \else
266       \expandafter\expandafter\expandafter\set@hyphenmins
267       \csname #1hyphenmins\endcsname\relax
268     \fi
269   \fi
270 }

271 \def\endhyphenrules{}
272 \def\providehyphenmins#1#2{%
273   \expandafter\ifx\csname #1hyphenmins\endcsname\relax

```

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 TeX works well anyway.

```

274 \namedef{#1hyphenmins}{#2}%
275 \fi}
276 \def\set@hyphenmins#1#2{\lefthyphenmin#1\righthyphenmin#2}
277 \def\LdfInit{%
278 \chardef\atcatcode=\catcode'\@
279 \catcode'\@=11\relax
280 \input babel.def\relax
281 \catcode'\@=\atcatcode \let\atcatcode\relax
282 \LdfInit}
283 \ifx\originalTeX\@undefined\let\originalTeX\@empty\fi
284 \ifx\babel@beginsave\@undefined\let\babel@beginsave\relax\fi
285 \ifx\PackageError\@undefined
286 \def\@nolanerr#1{%
287 \errhelp{Your command will be ignored, type <return> to proceed}%
288 \errmessage{You haven't defined the language #1\space yet}}
289 \def\@nopatterns#1{%
290 \message{No hyphenation patterns were loaded for}%
291 \message{the language '#1'}%
292 \message{I will use the patterns loaded for \string\language=0
293 instead}}
294 \def\@noopterr#1{%
295 \errmessage{The option #1 was not specified in \string\usepackage}
296 \errhelp{You may continue, but expect unexpected results}}
297 \def\@activated#1{%
298 \wlog{Package babel Info: Making #1 an active character}}
299 \else
300 \newcommand*\@nolanerr}[1]{%
301 \PackageError{babel}%
302 {You haven't defined the language #1\space yet}%
303 {Your command will be ignored, type <return> to proceed}}
304 \newcommand*\@nopatterns}[1]{%
305 \PackageWarningNoLine{babel}%
306 {No hyphenation patterns were loaded for\MessageBreak
307 the language '#1'\MessageBreak
308 I will use the patterns loaded for \string\language=0
309 instead}}
310 \newcommand*\@noopterr}[1]{%
311 \PackageError{babel}%
312 {You haven't loaded the option #1\space yet}%
313 {You may proceed, but expect unexpected results}}
314 \newcommand*\@activated}[1]{%
315 \PackageInfo{babel}{%
316 Making #1 an active character}}
317 \fi
318 \def\process@line#1#2 #3/{%
319 \ifx=#1
320 \process@synonym#2 /
321 \else
322 \process@language#1#2 #3/%
323 \fi

```



```

324 }
325 \toks@{}
326 \def\process@synonym#1 /{%
327   \ifnum\last@language=\m@ne
328     \expandafter\chardef\csname l@#1\endcsname\relax
329     \wlog{\string\l@#1=\string\language0}
330     \toks@\expandafter{\the\toks@
331       \expandafter\let\csname #1hyphenmins\expandafter\endcsname
332       \csname\language\hyphenmins\endcsname}%
333   \else
334     \expandafter\chardef\csname l@#1\endcsname\last@language
335     \wlog{\string\l@#1=\string\language\the\last@language}
336     \expandafter\let\csname #1hyphenmins\expandafter\endcsname
337     \csname\language\hyphenmins\endcsname
338   \fi
339 }
340 \def\process@language#1 #2 #3/{%
341   \expandafter\addlanguage\csname l@#1\endcsname
342   \expandafter\language\csname l@#1\endcsname
343   \def\language#1}%

```

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

```

344 \ifx\directlua@undefined
345   \global\toks8\expandafter{\the\toks8#1, }%
346 \else
347   \unless\ifcsname bbl@luatex@english@loaded\endcsname
348     \global\toks8\expandafter{\the\toks8#1 }%
349   \fi
350 \fi
351 \begingroup
352   \bbl@get@enc#1:\@@@
353   \ifx\bbl@hyph@enc\@empty
354   \else
355     \fontencoding{\bbl@hyph@enc}\selectfont
356   \fi
357   \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 `TEXby` 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.

```

358 \ifx\directlua@undefined
359   \input #2\relax
360 \else
361   \unless\ifcsname bbl@luatex@english@loaded\endcsname
362     \gdef\bbl@luatex@english@loaded{1}%
363     \input #2\relax

```

```

364     \fi
365     \fi
366     \ifnum\lefthyphenmin=\m@ne
367     \else
368         \expandafter\xdef\csname #1hyphenmins\endcsname{%
369             \the\lefthyphenmin\the\righthyphenmin}%
370     \fi
371 \endgroup
372 \ifnum\the\language=\z@
373     \expandafter\ifx\csname #1hyphenmins\endcsname\relax
374         \set@hyphenmins\tw@\thr@@\relax
375     \else
376         \expandafter\expandafter\expandafter\set@hyphenmins
377         \csname #1hyphenmins\endcsname
378     \fi
379     \the\toks@
380 \fi
381 \toks@{}%
382 \def\bbl@tempa{#3}%
383 \ifx\bbl@tempa\@empty
384 \else
385     \ifx\bbl@tempa\space
386     \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.

```

387     \ifx\directlua\@undefined
388         \input #3\relax
389     \fi
390     \fi
391 \fi
392 }
393 \def\bbl@get@enc#1:#2\@@@{%
394     \def\bbl@tempa{#1}%
395     \def\bbl@tempb{#2}%
396     \ifx\bbl@tempb\@empty
397         \let\bbl@hyph@enc\@empty
398     \else
399         \bbl@get@enc#2\@@@
400         \edef\bbl@hyph@enc{\bbl@tempa}%
401     \fi}
402 \openin1 = language.dat
403 \ifeof1
404     \message{I couldn't find the file language.dat,\space
405             I will try the file hyphen.tex}
406     \input hyphen.tex\relax
407 \else

```

```

408 \last@language\m@ne
409 \loop
410 \endlinechar\m@ne
411 \read1 to \bbl@line
412 \endlinechar'\^^M
413 \ifx\bbl@line\@empty
414 \else
415 \edef\bbl@line{\bbl@line\space/}%
416 \expandafter\process@line\bbl@line
417 \fi
418 \iftrue \csname fi\endcsname
419 \csname if\ifeof1 false\else true\fi\endcsname
420 \repeat
421 \language=0
422 \fi
423 \closein1
424 \let\process@language\@undefined
425 \let\process@synonym\@undefined
426 \let\process@line\@undefined
427 \let\bbl@tempa\@undefined
428 \let\bbl@tempb\@undefined
429 \let\bbl@eq\@undefined
430 \let\bbl@line\@undefined
431 \let\bbl@get@enc\@undefined
432 \ifx\addto@hook\@undefined
433 \else
434 \expandafter\addto@hook\expandafter\everyjob\expandafter{%
435 \expandafter\typeout\expandafter{\the\toks8 loaded.}}
436 \fi

```