The iflang package

Heiko Oberdiek*

2018/01/21 v1.7

Abstract

This package provides expandible checks for the current language based on macro \languagename or hyphenation patterns.

Contents

*Please report any issues at https://github.com/ho-tex/oberdiek/issues

1

1 Documentation

Package babel defines \iflanguagename. As first argument it takes a language name and executes the second or third argument depending on the current language. This language test is based on hypenation patterns. However, it is possible that different languages or dialects share the same patterns. In such cases \iflanguagename fails.

However, package babel and some other packages such as german or ngerman store the language name in the macro \languagename if \selectlanguage is called.

\IfLanguageName {⟨*lang*⟩} {⟨*then*⟩} {⟨*else*⟩}

Makro \IfLanguageName compares language ⟨*lang*⟩ with the current setting of macro \languagename. If both contains the same name then the ⟨*then*⟩ part is called, otherwise the ⟨*else*⟩ part.

The macro is expandable. Thus it can be safely used inside \edef or \csname. If case of errors like an undefined \languagename the ⟨*else*⟩ part is executed.

Note: Macro \IfLanguageName relies on the fact, that \languagename is set correctly:

Package babel:
Full support of \languagename in its language switching commands.

Format based on babel (language.dat):
If package babel is not used (or not yet loaded), then babel's hyphen.cfg has set \languagename to the last language in language.dat, but \language (current patterns) is zero and points to the first language. Thus the value of \languagename is basically garbage. Package iflang warns if \languagename and \language do not fit. This can be fixed by loading package babel previously.

Format based on ε-TEX's etex.src (language.def):
Unhappily it does not support \languagename. Thus this package hooks into \uselanguage to get \languagename defined and updated there. At package loading time the changed \uselanguage has not been called yet. Thus package iflang tries USenglish. This is the definite default language of etex.src. If the current patterns suit this default language, an undefined \languagename is set to this language. Otherwise a \languagename remains undefined and a warning is given.

\IfLanguagePatterns {⟨*lang*⟩} {⟨*then*⟩} {⟨*else*⟩}

This macro behaves similar to \IfLanguageName. But the language test is based on the current pattern in force (\language). Also this macro is expandable, in case of errors the ⟨*else*⟩ part is called.

The following naming convention for the pattern are supported:

babel/language.dat : \l@⟨*language*⟩

etex.src/language.def : \lang@⟨*language*⟩

Package iflang looks for \et@xpatterns (defined in etex.src) to find out the naming convention in use.

2 Implementation

1 ⟨∗package⟩

2.1 Reload check and package identification

Reload check, especially if the package is not used with LaTeX.

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
3   \catcode13=5 % ^^M
4   \endlinechar=13 %
5   \catcode35=6 % #
6   \catcode39=12 % '
7   \catcode44=12 % ,
8   \catcode45=12 % -
9   \catcode46=12 % .
10  \catcode58=12 % :
11  \catcode64=11 % @
12  \catcode123=1 % {
13  \catcode125=2 % }
14  \expandafter\let\expandafter\x\csname ver@iflang.sty\endcsname
15  \ifx\x\relax % plain-TeX, first loading
16  \else
17    \def\empty{}%
18    \ifx\x\empty % LaTeX, first loading,
19      % variable is initialized, but \ProvidesPackage not yet seen
20    \else
21      \expandafter\ifx\csname PackageInfo\endcsname\relax
22        \def\x#1#2{%
23          \immediate\write-1{Package #1 Info: #2.}%
24        }%
25      \else
26        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27      \fi
28      \x{iflang}{The package is already loaded}%
29      \aftergroup\endinput
30    \fi
31  \fi
32 \endgroup%
```

Package identification:

```
33 \begingroup\catcode61\catcode48\catcode32=10\relax%
34   \catcode13=5 % ^^M
35   \endlinechar=13 %
36   \catcode35=6 % #
37   \catcode39=12 % '
38   \catcode40=12 % (
39   \catcode41=12 % )
40   \catcode44=12 % ,
41   \catcode45=12 % -
42   \catcode46=12 % .
43   \catcode47=12 % /
44   \catcode58=12 % :
45   \catcode64=11 % @
46   \catcode91=12 % [
47   \catcode93=12 % ]
48   \catcode123=1 % {
49   \catcode125=2 % }
50   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
51     \def\x#1#2#3[#4]{\endgroup
```

```
52    \immediate\write-1{Package: #3 #4}%
53    \xdef#1{#4}%
54    }%
55  \else
56    \def\x#1#2[#3]{\endgroup
57      #2[{#3}]%
58      \ifx#1\@undefined
59        \xdef#1{#3}%
60      \fi
61      \ifx#1\relax
62        \xdef#1{#3}%
63      \fi
64    }%
65  \fi
66 \expandafter\x\csname ver@iflang.sty\endcsname
67 \ProvidesPackage{iflang}%
68    [2018/01/21 v1.7 Checks for the current language (HO)]%

69 \begingroup\catcode61\catcode48\catcode32=10\relax%
70  \catcode13=5 % ^^M
71  \endlinechar=13 %
72  \catcode123=1 % {
73  \catcode125=2 % }
74  \catcode64=11 % @
75  \def\x{\endgroup
76    \expandafter\edef\csname IfLang@AtEnd\endcsname{%
77      \endlinechar=\the\endlinechar\relax
78      \catcode13=\the\catcode13\relax
79      \catcode32=\the\catcode32\relax
80      \catcode35=\the\catcode35\relax
81      \catcode61=\the\catcode61\relax
82      \catcode64=\the\catcode64\relax
83      \catcode123=\the\catcode123\relax
84      \catcode125=\the\catcode125\relax
85    }%
86  }%
87 \x\catcode61\catcode48\catcode32=10\relax%
88 \catcode13=5 % ^^M
89 \endlinechar=13 %
90 \catcode35=6 % #
91 \catcode64=11 % @
92 \catcode123=1 % {
93 \catcode125=2 % }
94 \def\TMP@EnsureCode#1#2{%
95  \edef\IfLang@AtEnd{%
96    \IfLang@AtEnd
97    \catcode#1=\the\catcode#1\relax
98  }%
99  \catcode#1=#2\relax
100 }
101 \TMP@EnsureCode{39}{12}% '
102 \TMP@EnsureCode{40}{12}% (
103 \TMP@EnsureCode{41}{12}% )
104 \TMP@EnsureCode{44}{12}% ,
105 \TMP@EnsureCode{46}{12}% .
106 \TMP@EnsureCode{47}{12}% /
107 \TMP@EnsureCode{58}{12}% :
108 \TMP@EnsureCode{91}{12}% [
109 \TMP@EnsureCode{93}{12}% ]
```

```
110 \edef\IfLang@AtEnd{\IfLang@AtEnd\noexpand\endinput}
```

2.2 Tools

2.2.1 Provide some basic macros of LATEX

\@firstoftwo

```
111 \expandafter\ifx\csname @firstoftwo\endcsname\relax
112   \long\def\@firstoftwo#1#2{#1}%
113 \fi
```

\@secondoftwo

```
114 \expandafter\ifx\csname @secondoftwo\endcsname\relax
115   \long\def\@secondoftwo#1#2{#2}%
116 \fi
```

2.2.2 Expandible existence check for macros

\IfLang@IfDefined

```
117 \begingroup\expandafter\expandafter\expandafter\endgroup
118 \expandafter\ifx\csname ifcsname\endcsname\relax
119   \expandafter\@firstoftwo
120 \else
121   \expandafter\@secondoftwo
122 \fi
123 {%
124   \def\IfLang@IfDefined#1{%
125     \expandafter\ifx\csname#1\endcsname\relax
126       \expandafter\@secondoftwo
127     \else
128       \expandafter\@firstoftwo
129     \fi
130   }%
131 }{%
132   \def\IfLang@IfDefined#1{%
133     \ifnum\ifcsname#1\endcsname
134             \expandafter\ifx\csname#1\endcsname\relax
135               1%
136             \else
137               0%
138             \fi
139           \else
140             1%
141           \fi
142           =0 %
143       \expandafter\@firstoftwo
144     \else
145       \expandafter\@secondoftwo
146     \fi
147   }%
148 }
```

2.2.3 Macros for messages

```
149 \begingroup\expandafter\expandafter\expandafter\endgroup
150 \expandafter\ifx\csname RequirePackage\endcsname\relax
151   \input infwarerr.sty\relax
152   \input pdftexcmds.sty\relax
```

```
153 \else
154   \RequirePackage{infwarerr}[2007/09/09]%
155   \RequirePackage{pdftexcmds}[2016/05/16]%
156 \fi
```

2.2.4   Support for `etex.src`

`\IfLang@prefix`

```
157 \begingroup\expandafter\expandafter\expandafter\endgroup
158 \expandafter\ifx\csname et@xpatterns\endcsname\relax
159   \@PackageInfoNoLine{iflang}{%
160     Naming convention for patterns: babel%
161   }%
162   \def\IfLang@prefix{l@}%
163 \else
164   \@PackageInfoNoLine{iflang}{%
165     Naming convention for patterns: etex.src%
166   }%
167   \def\IfLang@prefix{lang@}%
168   \let\IfLang@OrgUseLanguage\uselanguage
169   \def\uselanguage#1{%
170     \edef\languagename{#1}%
171     \IfLang@OrgUseLanguage{#1}%
172   }%
```

The first `\uselanguage` that is executed as last line in `language.def` cannot patched this way. However, `language.def` is very strict. It forces the first added and used language to be USenglish. Thus, if `\languagename` is not defined, we can quite safely assume USenglish. As additional safety precaution the actual used patterns are checked.

```
173   \begingroup\expandafter\expandafter\expandafter\endgroup
174   \expandafter\ifx\csname languagename\endcsname\relax
175     \begingroup\expandafter\expandafter\expandafter\endgroup
176     \expandafter\ifx\csname lang@USenglish\endcsname\relax
177       \@PackageWarningNoLine{iflang}{%
178         \string\lang@USenglish\space is missing%
179       }%
180     \else
181       \ifnum\lang@USenglish=\language
182         \def\languagename{USenglish}%
183       \else
184         \@PackageWarningNoLine{iflang}{%
185           \string\languagename\space is not set,\MessageBreak
186           current language is unknown%
187         }%
188       \fi
189     \fi
190   \fi
191 \fi
192 \begingroup\expandafter\expandafter\expandafter\endgroup
193 \expandafter\ifx\csname languagename\endcsname\relax
194   \@PackageInfoNoLine{iflang}{%
195     \string\languagename\space is not set%
196   }%
197 \fi
```

2.3 \IfLanguagePatterns

```
198 \def\IfLanguagePatterns#1{%
199   \ifnum\IfLang@IfDefined{\IfLang@prefix#1}{%
200         \ifnum\csname\IfLang@prefix#1\endcsname=\language
201           0%
202         \else
203           1%
204         \fi
205      }{1}=0 %
206     \expandafter\@firstoftwo
207   \else
208     \expandafter\@secondoftwo
209   \fi
210 }
```

2.4 \IfLanguageName

```
211 \begingroup\expandafter\expandafter\expandafter\endgroup
212 \expandafter\ifx\csname pdf@strcmp\endcsname\relax
213   \expandafter\@firstoftwo
214 \else
215   \expandafter\@secondoftwo
216 \fi
217 {%
```

We do not have \pdf@strcmp (and \pdfstrcmp). Thus we must define our own expandable string comparison. The following implementation is based on a TEX pearl from David Kastrup, presented at the conference BachoTEX 2005: http://www.gust.org.pl/projects/pearls/2005p/david-kastrup/bachotex2005-david-kastrup-pearl1.pdf

The orignal code allows macros inside the second string. Because also \languagename might consists of further macros, we need a variant that allows macros in the first string, too.

```
218   \def\IfLang@StrNil{\relax}%
219   \def\IfLang@StrEqual#1{%
220     \number\IfLang@StrEqualStart{}{}#1\IfLang@StrNil
221   }%
222   \def\IfLang@StrEqualStart#1#2#3{%
223     \ifx#3\IfLang@StrNil
224       \IfLang@StrEqualStop
225     \fi
226     \ifcat\noexpand#3\relax
227       \IfLang@StrExpand{#1}{#2}#3%
228     \fi
229     \IfLang@StrEqualStart{\if#3#1}{#2\fi}%
230   }%
231   \def\IfLang@StrEqualStop\fi#1\IfLang@StrEqualStart#2#3#4{%
232     \fi
233     #2#4\relax'#313 %
234   }%
235   \def\IfLang@StrExpand#1#2#3\fi\IfLang@StrEqualStart#4#5{%
236     \fi
237     \IfLang@@StrExpand{#1}{#2}#3%
238   }%
239   \def\IfLang@@StrExpand#1#2#3\IfLang@StrNil{%
240     \expandafter\IfLang@@@StrExpand#3\IfLang@StrNil{#1}{#2}%
```

```
241    }%
242    \def\IfLang@@@StrExpand#1\IfLang@StrNil#2#3{%
243      \IfLang@StrEqualStart{#2}{#3}#1\IfLang@StrNil
244    }%
```

\IfLanguageName

```
245    \def\IfLanguageName#1{%
246      \ifnum\IfLang@IfDefined{languagename}{%
247          \if\expandafter\IfLang@StrEqual\expandafter%
248                        {\languagename}{#1}%
249            0%
250          \else
251            1%
252          \fi
253        }{1}=0 %
254      \expandafter\@firstoftwo
255    \else
256      \expandafter\@secondoftwo
257    \fi
258    }%

259 }{%
```

\IfLanguageName

```
260    \def\IfLanguageName#1{%
261      \ifnum\IfLang@IfDefined{languagename}{%
262          \pdf@strcmp{#1}{\languagename}%
263        }{1}=0 %
264      \expandafter\@firstoftwo
265    \else
266      \expandafter\@secondoftwo
267    \fi
268    }%

269 }
```

2.5   Check plausibility of \languagename

```
270 \begingroup\expandafter\expandafter\expandafter\endgroup
271 \expandafter\ifx\csname languagename\endcsname\relax
272 \else
273   \IfLanguagePatterns{\languagename}{}{%
274     \@PackageWarningNoLine{iflang}{%
275       Mismatch between \string\language\space
276       (patterns)\MessageBreak
277       and setting of \string\languagename
278     }%
279   }%
280 \fi

281 \IfLang@AtEnd%
282 ⟨/package⟩
```

3   Installation

3.1   Download

Package.   This package is available on CTAN[1]:

---

[1] CTAN:pkg/iflang

CTAN:macros/latex/contrib/oberdiek/iflang.dtx The source file.

CTAN:macros/latex/contrib/oberdiek/iflang.pdf Documentation.

Bundle.   All the packages of the bundle 'oberdiek' are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

CTAN:install/macros/latex/contrib/oberdiek.tds.zip

TDS refers to the standard "A Directory Structure for TeX Files" (CTAN:pkg/tds). Directories with `texmf` in their name are usually organized this way.

3.2   Bundle installation

Unpacking.   Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

3.3   Package installation

Unpacking.   The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain TeX:

```
tex iflang.dtx
```

TDS.   Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

```
iflang.sty → tex/generic/oberdiek/iflang.sty
iflang.pdf → doc/latex/oberdiek/iflang.pdf
iflang.dtx → source/latex/oberdiek/iflang.dtx
```

If you have a `docstrip.cfg` that configures and enables `docstrip`'s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

3.4   Refresh file name databases

If your TeX distribution (TeX Live, MiKTeX, …) relies on file name databases, you must refresh these. For example, TeX Live users run `texhash` or `mktexlsr`.

3.5   Some details for the interested

Unpacking with LaTeX.   The `.dtx` chooses its action depending on the format:

plain TeX: Run `docstrip` and extract the files.

LaTeX: Generate the documentation.

If you insist on using LaTeX for `docstrip` (really, `docstrip` does not need LaTeX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{iflang.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfLaTeX:

```
pdflatex iflang.dtx
makeindex -s gind.ist iflang.idx
pdflatex iflang.dtx
makeindex -s gind.ist iflang.idx
pdflatex iflang.dtx
```

4 Acknowledgement

I wish to thank:

Markus Kohm Useful hints for version 1.2.

5 History

[2007/04/10 v1.0]

- First public version.

[2007/04/11 v1.1]

- Line ends sanitized.

[2007/04/12 v1.2]

- Initialization of `\languagename` in case of `etex.src`.
- Some sanity tests added.
- Documentation improved.

[2007/04/26 v1.3]

- Use of package infwarerr.

[2007/09/09 v1.4]

- Bug fix: `\IfLang@StrEqual` → `\IfLangStrEqual` (Gabriele Balducci).
- Catcode section rewritten.

[2007/11/11 v1.5]

- Use of package pdftexcmds for LuaTeX support.

[2016/05/16 v1.6]

- Documentation updates.

[2018/01/21 v1.7]

- Fix test for `etex.src`.

6   Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.