The \texttt{iflang} package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/11/11 v1.5

Abstract

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

Contents

1 Documentation 2

2 Implementation 2
  2.1 Reload check and package identification 3
  2.2 Tools 4
    2.2.1 Provide some basic macros of \LaTeX 4
    2.2.2 Expandible existence check for macros 4
    2.2.3 Macros for messages 5
    2.2.4 Support for \texttt{etex.src} 5
  2.3 \texttt{IfLanguagePatterns} 6
  2.4 \texttt{IfLanguageName} 6
  2.5 Check plausibility of \texttt{languagename} 7

3 Test 8
  3.1 Catcode checks for loading 8
  3.2 Test with \LaTeX 9
  3.3 Test with plain-\TeX and $\varepsilon$-\TeX 10
  3.4 Test with plain-\TeX and without $\varepsilon$-\TeX/pdf\TeX 11

4 Installation 12
  4.1 Download 12
  4.2 Bundle installation 12
  4.3 Package installation 13
  4.4 Refresh file name databases 13
  4.5 Some details for the interested 13

5 Acknowledgement 14

6 History 14
  [2007/04/10 v1.0] 14
  [2007/04/11 v1.1] 14
  [2007/04/12 v1.2] 14
  [2007/04/26 v1.3] 14
  [2007/11/11 v1.5] 14

7 Index 14
1 Documentation

Package babel defines \iflanguage. 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 hyphenation patterns. However, it is possible that different languages or dialects share the same patterns. In such cases \iflanguage 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-TX’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 \uselanguage (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 \TeX.

\begingroup
\catcode44 12 \% ,
\catcode45 12 \% -
\catcode46 12 \%
\catcode48 12 \%
\catcode49 11 \%
\catcode123 1 \%
\catcode125 2 \%
\expandafter\let\expandafter\x\csname ver@iflang.sty\endcsname
\ifx\x\relax \else
\def\empty{}\fi
\ifx\empty \fi
\catcode35 6 \% #
\catcode40 12 \% (}
\catcode41 12 \% )
\catcode44 12 \%
\catcode45 12 \% -
\catcode46 12 \%
\catcode47 12 \% /
\catcode48 12 \%
\catcode49 12 \%
\catcode91 12 \%
\catcode93 12 \%
\expandafter\ifx\csname ProvidesPackage\endcsname\relax
\def\x#1#2[#3]{\endgroup
\immediate\write-1{Package \#1 Info: \#2.}}\%
\x{iflang}{The package is already loaded}%
\aftergroup\endinput
\fi
\fi
\endgroup

Package identification:

\begingroup
\catcode35 6 \% #
\catcode40 12 \% (}
\catcode41 12 \% )
\catcode44 12 \%
\catcode45 12 \% -
\catcode46 12 \%
\catcode47 12 \% /
\catcode48 12 \%
\catcode49 12 \%
\catcode91 12 \%
\catcode93 12 \%
\expandafter\ifx\csname ProvidesPackage\endcsname\relax
\def\x#1#2[#3]{\endgroup
\immediate\write-1{Package \#1 #3}}%
\x{iflang}{The package is already loaded}%
\aftergroup\endinput
\fi
\fi
\endgroup
\providepackage{iflang}[
[2007/11/11 v1.5 Language checks (HO)]

\begingroup
\catcode123 1 % {
\catcode125 2 % }
\def\x{\endgroup
\expandafter\edef\csname IfLang@AtEnd\endcsname{%
\catcode35 \the\catcode35\relax
\catcode64 \the\catcode64\relax
\catcode123 \the\catcode123\relax
\catcode125 \the\catcode125\relax
}%
}
\x
\catcode35 6 % #
\catcode64 11 % @
\catcode123 1 % {
\catcode125 2 % }
\def\TMPEncode#1#2{%
\edef\IfLang@AtEnd{%
\IfLang@AtEnd\catcode#1 \the\catcode#1\relax
}\catcode#1 #2\relax
}%
\\TMPEncode{39}{12}% '
\\TMPEncode{40}{12}% (
\\TMPEncode{41}{12}% )
\\TMPEncode{44}{12}% ,
\\TMPEncode{46}{12}% .
\\TMPEncode{47}{12}% /
\\TMPEncode{58}{12}% :
\\TMPEncode{61}{12}% =

\section{Tools}

\subsection{Provide some basic macros of LATEX}

\@firstoftwo
\expandafter\if\csname @firstoftwo\endcsname\relax
\long\def\@firstoftwo#1#2(#1)%
\fi

\@secondoftwo
\expandafter\if\csname @secondoftwo\endcsname\relax
\long\def\@secondoftwo#1#2(#2)%
\fi

\subsection{Expandible existence check for macros}

\IfLang@ifDefined
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\if\csname \ifcsname\endcsname\relax
\expandafter\@firstoftwo
\else
\expandafter\@secondoftwo
\fi
\%}
\def\IfLang@ifDefined#1{%
\expandafter\if\csname\endcsname\relax

2.2.3 Macros for messages
\begin{verbatim}
\def\IfLang@ifDefined#1{%
  \ifnum\ifcsname#1\endcsname
    1%
    \else
      0%
  \fi
  \fi
}\end{verbatim}

2.2.4 Support for etex.src
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 \texttt{USenglish}. Thus, if \texttt{languagename} is not defined, we
can quite safely assume \texttt{USenglish}. As additional safety precaution the actual
used patterns are checked.
\begin{verbatim}
\def\IfLang@prefix{l@}\
\let\IfLang@OrgUseLanguage\uselanguage
\def\uselanguage#1{%
  \edef\languagename{#1}
  \IfLang@OrgUseLanguage{#1}
}\end{verbatim}
\IfLanguagePatterns

2.3 \IfLanguagePatterns

\IfLanguagePatterns
\def\IfLanguagePatterns#1{% 
\ifnum\IfLang@IfDefined{\IfLang@prefix#1}{% 
\ifnum\csname\IfLang@prefix#1\endcsname=\language 
0% 
\else 
1% 
\fi 
}\{i\}=0 % 
\expandafter\@firstoftwo 
\else 
\expandafter\@secondoftwo 
\fi 
}

2.4 \IfLanguageName

\IfLanguageName

We do not have \pdf@strcmp (and \pdfstrcmp). Thus we must define our own expandable string comparison. The following implementation is based on a Te\TeX
derived from David Kastrup, presented at the conference Bacho\TeX

The original 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.

\def\IfLang@StrNil{\relax}%
\def\IfLang@StrEqual#1{%
\number\IfLang@StrEqualStart{}#1\IfLang@StrNil
2.5 Check plausibility of $\textsc{language}$name

\begin{verbatim}
\IfLanguageName#1{
  \ifnum\IfLang@IfDefined{languagename}{% 
    \if\expandafter\IfLang@StrEqual\expandafter{
      \languagename}{#1}0 \else 1 \fi
  \else \expandafter\@firstoftwo \fi
  \expandafter\@secondoftwo
\}\% 

\IfLanguageName#1{% 
  \ifnum\IfLang@IfDefined{languagename}{% 
    pdf@strcmp(#1)(\languagename)0 \else \expandafter\@firstoftwo \fi
    \expandafter\@secondoftwo
  \}\% 

begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname languagename\endcsname\relax
  \else \IfLanguagePatterns{\languagename}{}{%
  @PackageWarningNoLine{iflang}{Mismatch between \string\textsc{language}\space space}
\end{verbatim}

\end{verbatim}
patterns\MessageBreak
and setting of \string\languagename
}
\fi
\IfLangAtEnd
⟨/package⟩

3  Test

3.1  Catcode checks for loading

⟨∗\test1⟩
\catcode'\{=1
\catcode'\}=2
\catcode'\#=6
\catcode'@=11
\expandafter\ifx\csname count@\endcsname\relax
\countdef\count@=255
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
\long\def\@gobble#1{}
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
\long\def\@firstofone#1{#1}
\fi
\expandafter\if\csname loop\endcsname\relax
\expandafter\@firstofone
\else
\expandafter\@gobble
\fi
{%
\def\loop#1\repeat{%
\def\body{#1}%
\iterate
%
\def\iterate{%
\body
\let\next\iterate
\else
\let\next\relax
\fi
\next
%
\let\repeat=\fi
%
\def\RestoreCatcodes{}%}
\count@=0
\loop
\edef\RestoreCatcodes{%
\RestoreCatcodes
\catcode\the\count@=\the\catcode\count@\relax
}
\ifnum\count@<255
\advance\count@ 1
\repeat
\def\RangeCatcodeInvalid#1#2{%
\count@=#1\relax
\loop
\catcode\the\count@=\the\catcode\count@\relax
}
\ifnum\count@<255
\advance\count@ 1
\repeat
\def\RangeCatcodeInvalid#1#2{%
\count@=#1\relax
\loop
\catcode\count@=15
\ifnum\count@<#2\relax
8
3.2 Test with \LaTeX

\NeedsTeXFormat{LaTeX2e}
\documentclass{minimal}
\usepackage{qstest}
\IncludeTests{*}
\LogTests{log}{*}{*}
\usepackage[english,naustrian,ngerman]{babel}
\usepackage{iflang}
\begin{document}
\begin{qstest}{IfLanguagePatterns}{language, pattern}
\def\test#1#2{\Expect*{\IfLanguagePatterns{#1}{true}{false}}{#2}}
\test{ngerman}{true}
\test{naustrian}{true}
\test{english}{false}
\test{foobar}{false}
\end{qstest}
\begin{qstest}{IfLanguageName}{language, name}
\def\test#1#2{\Expect*{\IfLanguageName{#1}{true}{false}}{#2}}
\test{ngerman}{true}
\test{naustrian}{false}
\selectlanguage{naustrian}
\test{ngerman}{false}
\test{naustrian}{true}
\test{foobar}{false}
\end{qstest}
\end{document}
\input iflang.sty
\catcode64=12
\def\TestGeneric#1#2#3{%\begingroup \edef\x{#1{#2}{true}{false}}\edef\y{#3}\ifx\x\y\else\errmessage{Failed test: \string#1{#2} <> #3}\fi\endgroup}
\makeatletter
\begin{qstest}{IfDefined}{defined}
\let\foobar\relax
\Expect*{\IfLang@IfDefined{foobar}{true}{false}}{false}%
\Expect*{\ifx\foobar\relax true\else false\fi}{true}%
\let\foobar\UNDEFINED
\Expect*{\IfLang@IfDefined{foobar}{true}{false}}{false}%
\Expect*{\ifx\foobar\UNDEFINED true\else false\fi}{false}%
\Expect*{\ifx\foobar\UNDEFINED true\else false\fi}{true}%
\end{qstest}
\end{document}

3.3 Test with plain-\TeX{} and \$\varepsilon$-\TeX{}
\begin{verbatim}
\%\% Format ‘etex’ based on ‘language.def’
\input iflang.sty
\catcode64=12
\def\TestGeneric#1#2#3{%\begingroup \edef\x{#1{#2}{true}{false}}\edef\y{#3}\ifx\x\y\else\errmessage{Failed test: \string#1{#2} <> #3}\fi\endgroup}
\makeatletter
\begin{qstest}{IfDefined}{defined}
\let\foobar\relax
\Expect*{\IfLang@IfDefined{foobar}{true}{false}}{false}%
\Expect*{\ifx\foobar\relax true\else false\fi}{true}%
\let\foobar\UNDEFINED
\Expect*{\IfLang@IfDefined{foobar}{true}{false}}{false}%
\Expect*{\ifx\foobar\UNDEFINED true\else false\fi}{false}%
\Expect*{\ifx\foobar\UNDEFINED true\else false\fi}{true}%
\end{qstest}
\end{document}
\end{verbatim}
\def\TestPatterns\{\TestGeneric\IfLanguagePatterns\}
\def\TestName\{\TestGeneric\IfLanguageName\}
\TestPatterns\{USenglish\}{true}
\TestPatterns\{ngerman\}{false}
\TestName\{USenglish\}{true}
\TestName\{ngerman\}{false}
\uselanguage\{ngerman\}
\TestPatterns\{USenglish\}{false}
\TestPatterns\{ngerman\}{true}
\TestName\{USenglish\}{false}
\TestName\{ngerman\}{true}
\csname @@end\endcsname
\end
⟨/test4⟩

3.4 Test with plain-TEX and without $ɛ$-TEX/pdfTEX

⟨∗test5⟩
%% Format 'tex' (vanilla plain-TEX)
\let\ifcsname\UNDEFINED
\let\pdfstrcmp\UNDEFINED
\input iflang.sty
\catcode64=11
\def\TestDefined#1{\
\IfLang@IfDefined{foobar}\{\}\%\n\if\foobart#1\%
\else\n\errmessage{Failed test: \string\foobar <> \string#1}\%
\fi
\}
\let\foobar\relax
\TestDefined\relax
\let\foobar\UNDEFINED
\TestDefined\relax
\beginpgroup
\edef\x\{%\n\if\IfLang@StrEqual{#1}{#2}\%
true\%
\else\nfalse\%
\fi
\%
\def\expect\{#3\%
\if\x\expect\%
\else\n\def\a\{#1\%
\@onelevel@sanitize\a
\def\b\{#2\%
\@onelevel@sanitize\b
\errmessage{Failed test: ‘\a’=’\b’ <> \expect}\%
\edef\n\{#1\%
\fi
\endgroup
\TestCompare{junk}{junk}{true}
\TestCompare{}{}{true}
\TestCompare{a}{b}{false}
\TestCompare{aa}{bb}{false}
def\a{ax}
def\b{bx}
def\c{\a\b}
def\d{\c\b}
def\exch#1#2{#2#1}
def\gobble#1{}
\TestCompare{\gobble a}{}{true}
\TestCompare{}{\gobble a}{true}
\TestCompare{a\exch xyb}{ayxb}{true}
\TestCompare{\c}{\c}{true}
\TestCompare{\d}{\c\b}{true}
\csname @@end\endcsname
\end
(/test5)

4 Installation

4.1 Download
Package. This package is available on CTAN:\footnote{ftp://ftp.ctan.org/tex-archive/}


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 docu-
mentation 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:tds/
tds.pdf). Directories with texmf in their name are usually organized this way.

4.2 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

Script installation. Check the directory TDS:scripts/oberdiek/ for scripts
that need further installation steps. Package attachfile2 comes with the Perl script
pdfatfi.pl that should be installed in such a way that it can be called as pdfatfi.
Example (linux):

chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
4.3 Package installation

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

\texttt{tex \textbackslash{} iflang.dtx}

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

\begin{verbatim}
iflang.sty → tex/generic/oberdiek/iflang.sty
iflang.pdf → doc/latex/oberdiek/iflang.pdf
test/iflang-test1.tex → doc/latex/oberdiek/test/iflang-test1.tex
test/iflang-test2.tex → doc/latex/oberdiek/test/iflang-test2.tex
test/iflang-test3.tex → doc/latex/oberdiek/test/iflang-test3.tex
test/iflang-test4.tex → doc/latex/oberdiek/test/iflang-test4.tex

\texttt{test/iflang-test5.tex → doc/latex/oberdiek/test/iflang-test5.tex}
iflang.dtx → source/latex/oberdiek/iflang.dtx
\end{verbatim}

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

4.4 Refresh file name databases

If your T\TeX{} distribution (te\TeX{}, mik\TeX{}, ...) relies on file name databases, you must refresh these. For example, te\TeX{} users run \texttt{texhash} or \texttt{mktexlsr}.

4.5 Some details for the interested

Attached source. The PDF documentation on CTAN also includes the .dtx source file. It can be extracted by AcrobatReader 6 or higher. Another option is \texttt{pdftk}, e.g. unpack the file into the current directory:

\texttt{pdftk iflang.pdf unpack\_files output .}

Unpacking with \LaTeX{}. The .dtx chooses its action depending on the format:

\begin{description}
\item[plain-T\TeX{}] Run docstrip and extract the files.
\item[\LaTeX{}] Generate the documentation.
\end{description}

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

\texttt{latex \textbackslash{}let\textbackslash{}install=y\textbackslash{}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 \texttt{ltxdoc.cfg}. For instance, put this line into this file, if you want to have A4 as paper format:

\begin{verbatim}
\PassOptionsToClass{a4paper}{article}
\end{verbatim}

An example follows how to generate the documentation with \texttt{pdflatex}:

\begin{verbatim}
pdf\textbackslash{}latex iflang.dtx
make\textbackslash{}index -s gind.ist iflang.idx
pdf\textbackslash{}latex iflang.dtx
make\textbackslash{}index -s gind.ist iflang.idx
pdf\textbackslash{}latex iflang.dtx
\end{verbatim}
5 Acknowledgement
I wish to thank:
Markus Kohm Useful hints for version 1.2.

6 History

[2007/04/10 v1.0]
• First public version.

[2007/04/11 v1.1]
• Line ends sanitized.

[2007/04/12 v1.2]
• Initialization of \language 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 L\TeX support.

7 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; numbers in roman refer to the code lines where the entry is used.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>{</th>
<th>267, 329</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>269, 331</td>
<td></td>
</tr>
<tr>
<td>%</td>
<td>334</td>
<td></td>
</tr>
<tr>
<td>@</td>
<td>270, 327</td>
<td></td>
</tr>
<tr>
<td>@PackageInfoNoLine</td>
<td>142, 147, 177</td>
<td></td>
</tr>
<tr>
<td>@PackageWarningNoLine</td>
<td>160, 167, 257</td>
<td></td>
</tr>
<tr>
<td>@firstofone</td>
<td>278, 281</td>
<td></td>
</tr>
<tr>
<td>@firstoftwo</td>
<td>94, 102, 111, 126, 189, 196, 237, 247</td>
<td>275, 283, 328, 332, 345, 395, 409, 491, 492, 495</td>
</tr>
<tr>
<td>@gobble</td>
<td></td>
<td>267, 329</td>
</tr>
<tr>
<td>@onelevel@sanitize</td>
<td>385, 476, 492, 494</td>
<td>308, 316, 354, 355, 364, 409</td>
</tr>
<tr>
<td>@seconddoftwo</td>
<td>97, 104, 109, 128, 191, 198, 239, 249</td>
<td></td>
</tr>
<tr>
<td>@undefined</td>
<td>52</td>
<td>149, 205</td>
</tr>
<tr>
<td>[</td>
<td>332</td>
<td></td>
</tr>
<tr>
<td>\</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>\a</td>
<td>491, 492, 495, 503, 505</td>
<td>A</td>
</tr>
<tr>
<td>\advance</td>
<td>308, 316</td>
<td></td>
</tr>
<tr>
<td>\aftergroup</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>\b</td>
<td>493, 494, 495, 504, 505, 506, 513</td>
<td>B</td>
</tr>
<tr>
<td>\begin</td>
<td>354, 355, 364, 409</td>
<td></td>
</tr>
<tr>
<td>\body</td>
<td>287, 291</td>
<td></td>
</tr>
</tbody>
</table>