Abstract

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

Contents

1 Documentation

2 Implementation

2.1 Reload check and package identification

2.2 Tools

2.2.1 Provide some basic macros of \LaTeX

2.2.2 Expandible existence check for macros

2.2.3 Macros for messages

2.2.4 Support for \texttt{etex.src}

2.3 \texttt{IfLanguagePatterns}

2.4 \texttt{IfLanguageName}

2.5 Check plausibility of \texttt{languagename}

3 Test

3.1 Catcode checks for loading

3.2 Test with \LaTeX

3.3 Test with plain \TeX and \texttt{e-\TeX}

3.4 Test with plain \TeX and without \texttt{e-\TeX/pdf\TeX}

4 Installation

4.1 Download

4.2 Bundle installation

4.3 Package installation

4.4 Refresh file name databases

4.5 Some details for the interested

5 Catalogue

6 Acknowledgement

7 History

[2007/04/10 v1.0]

[2007/04/11 v1.1]

[2007/04/12 v1.2]

[2007/04/26 v1.3]

[2007/09/09 v1.4]

[2007/11/11 v1.5]

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

Package \texttt{babel} defines \texttt{\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 \texttt{\iflanguage} fails.

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

\begin{verbatim}
\IfLanguageName {⟨lang⟩} {⟨then⟩} {⟨else⟩}
\end{verbatim}

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

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

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

Package \texttt{babel}:

Full support of \texttt{\languagename} in its language switching commands.

Format based on babel (\texttt{language.dat}): 

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

Format based on \LaTeX{}'s \texttt{etex.src} (\texttt{language.def}): 

Unhappily it does not support \texttt{\languagename}. Thus this package hooks into \texttt{\uselanguage} to get \texttt{\languagename} defined and updated there. At package loading time the changed \texttt{\uselanguage} has not been called yet. Thus package \texttt{iflang} tries \texttt{USenglish}. This is the definite default language of \texttt{etex.src}. If the current patterns suit this default language, an undefined \texttt{\languagename} is set to this language. Otherwise a \texttt{\languagename} remains undefined and a warning is given.

\begin{verbatim}
\IfLanguagePatterns {⟨lang⟩} {⟨then⟩} {⟨else⟩}
\end{verbatim}

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

The following naming convention for the pattern are supported:

\texttt{babel/language.dat} : \texttt{\l@⟨language⟩}

\texttt{etex.src/language.def} : \texttt{\lang@⟨language⟩}

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

2.1 Reload check and package identification

Reload check, especially if the package is not used with \LaTeX.\n
\begin{verbatim}
\expandafter\let\expandafter\x\csname ver@iflang.sty\endcsname
\ifx\x\relax % plain-TeX, first loading
\else
\expandafter\ifx\csname PackageInfo\endcsname\relax
\def\x#1#2{\immediate\write-1{Package #1 Info: #2, stopped}}\%
\else
\def\x#1#2{\PackageInfo{#1}{#2, stopped}}\%
\fi
\x{iflang}{The package is already loaded}\
\aftergroup\endinput
\fi
\fi
\endgroup%
\end{verbatim}

Package identification:

\begin{verbatim}
\expandafter\let\expandafter\x\csname ver@iflang.sty\endcsname
\ifx\x\relax % plain-TeX, first loading
\else
\expandafter\ifx\csname ProvidesPackage\endcsname\relax
\def\x#1#2#3[#4]{\endgroup\immediate\write-1{Package: #3 #4}\%}
\xdef#1{#4}\
\else
\def\x#1#2[#3]{\endgroup\immediate\write-1{Package #1 (#2, stopped)}\%}
\fi
\fi
\endgroup%
\end{verbatim}
2.2 Tools

2.2.1 Provide some basic macros of \LaTeX

\@firstoftwo
2.2.2 Expandible existence check for macros

2.2.3 Macros for messages

2.2.4 Support for etex.src
\else
\PackageInfoNoLine@iflang{%
Naming convention for patterns: \etex\src%
}\}
\def\IfLang@prefix{lang@}%
\let\IfLang@OrgUseLanguage\uselanguage
\def\uselanguage#1{%
\edef\languagename{#1}%
\IfLang@OrgUseLanguage{#1}%
}%

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.

\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname languagename\endcsname\relax
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname lang@USenglish\endcsname\relax
\PackageWarningNoLine@iflang{%
\string\lang@USenglish\space is missing%
}%
\else
\@@PackageWarningNoLine@iflang{%
\string\languagename\space is not set,\MessageBreak
\text{text\current language is unknown}%
}%
\fi
\fi
\fi
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname languagename\endcsname\relax
\PackageInfoNoLine@iflang{%
\string\languagename\space is not set%
}\fi
\fi
\endgroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifnum\csname lang@USenglish\endcsname=\language
\def\languagename{USenglish}%
\else
\@@PackageWarningNoLine@iflang{%
\string\languagename\space is not set%
}\fi
\fi
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifnum\csname lang@USenglish\endcsname=\language
\PackageInfoNoLine@iflang{%
\string\languagename\space is not set%
}\fi
\fi

\section*{2.3 \texttt{\IfLanguagePatterns}}

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

\section*{2.4 \texttt{\IfLanguageName}}
We do not have \pdfstrcmp (and \pdfstricomp). 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 Bacho\TeX 2005: http://www-stary.gust.org.pl/pearls/2005/david-kastrup/bachotex2005-david-kastrup-pearl1.pdf

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\IfLanguageName#1{% 
  \ifnum\IfLang@IfDefined{languagename}{% 
    \if\expandafter\IfLang@StrEqual\expandafter{% 
      \languagename{% #1}{% 
        0% 
      }{% #2}% 
    }{% #2}=% 
  }{% #2}=0% 
  \expandafter\@firstoftwo 
  \else 
    \expandafter\@secondoftwo 
  \fi 
}
2.5 Check plausibility of \texttt{languagename}

3 Test

3.1 Catcode checks for loading
\edef\RangeCatcodeInvalid#1#2{\
\count@=#1\relax\
\loop\
\ifnum\count@<#2\relax\
\advance\count@ 1 %
\repeat\
\errmessage{Character \the\count@ space with wrong catcode \the\catcode\count@ space instead of \number#3%}
\}
\def\RangeCatcodeCheck#1#2#3{\
\count@=#1\relax\
\loop\
\ifnum#3=\catcode\count@
\else
\errmessage{Character \the\count@ space with wrong catcode \the\catcode\count@ space instead of \number#3%}
\fi
\ifnum\count@<#2\relax\
\advance\count@ 1 %
\repeat
\}
\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
\def\LoadCommand{\input iflang.sty}\relax
\fi
\def\Test{\
\RangeCatcodeInvalid{0}{47}\%
\RangeCatcodeInvalid{58}{64}\%
\RangeCatcodeInvalid{91}{96}\%
\RangeCatcodeInvalid{123}{255}\%
\catcode`\@=12 %
\catcode`\\=0 %
\catcode`\%=14 %
\LoadCommand
\RangeCatcodeCheck{0}{36}{15}\%
\RangeCatcodeCheck{37}{37}{14}\%
\RangeCatcodeCheck{48}{47}{15}\%
\RangeCatcodeCheck{8}{57}{12}\%
\RangeCatcodeCheck{58}{63}{15}\%
\RangeCatcodeCheck{64}{64}{12}\%
\RangeCatcodeCheck{65}{90}{11}\%
\RangeCatcodeCheck{91}{91}{15}\%
\RangeCatcodeCheck{92}{92}{0}\%
\RangeCatcodeCheck{93}{96}{15}\%
\RangeCatcodeCheck{97}{122}{11}\%
\}
3.2 Test with \LaTeX

\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}

\begin{document}
\end{document}
\begin{qstest}{IfDefined}{defined}
\makeatletter
\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\relax 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{test4}
\%\% 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}
\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
3.4 Test with plain \TeX{} and without \$\varepsilon$-\TeX{}/pdf\TeX{}

\begin{verbatim}
\%\% Format `tex' (vanilla plain-Tex)
\let\ifcsname\UNDEFINED
\def\nifdefefined{0}
\let\ifdefefined\nifdefined
\let\ifx\UNDEFINED
\def\exactdef{1}
\let\exactdef\UNDEFINED
\input iflang.sty
\catcode64=11
\def\TestDefined#1{%
  \IfLang@IfDefined{foobar}{}{}%
  \ifx\foobar#1%
  \else
    \errmessage{Failed test: \string\foobar <> \string#1}%
  \fi
  }
\let\foobar\relax
\TestDefined\relax
\let\foobar\UNDEFINED
\TestDefined\relax
\def\strip@prefix#1>{}
\def\@onelevel@sanitize#1{%
  \edef#1{\expandafter\strip@prefix\meaning#1}%
}
\def\TestCompare#1#2#3{%
  \begingroup
  \edef\x{\IfLang@StrEqual{#1}{#2}true \else false\fi}
  \def\expect{#3}
  \ifx\x\expect
  \else
    \def\a{#1}
    \@onelevel@sanitize\a
    \def\b{#2}
    \@onelevel@sanitize\b
    \errmessage{Failed test: `\a'='\b' <> \expect}%
  \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}
\TestCompare{\d\c}{\c\b}{true}
\end{verbatim}

\end{verbatim}
4 Installation

4.1 Download

Package. This package is available on CTAN\textsuperscript{1}:

\texttt{CTAN:macros/latex/contrib/oberdiek/iflang.dtx} The source file.


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.

\texttt{CTAN:install/macros/latex/contrib/oberdiek.tds.zip}

\textit{TDS} refers to the standard “A Directory Structure for \TeX{} Files” (\texttt{CTAN:tds/tds.pdf}). Directories with \texttt{texmf} in their name are usually organized this way.

4.2 Bundle installation

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

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

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

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

4.3 Package installation

Unpacking. The \texttt{.dtx} file is a self-extracting docstrip archive. The files are extracted by running the \texttt{.dtx} through \texttt{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 \texttt{texmf} tree):

```
iflang.sty \rightarrow tex/generic/oberdiek/iflang.sty
iflang.pdf \rightarrow doc/latex/oberdiek/iflang.pdf
test/iflang-test1.tex \rightarrow doc/latex/oberdiek/test/iflang-test1.tex
test/iflang-test2.tex \rightarrow doc/latex/oberdiek/test/iflang-test2.tex
test/iflang-test3.tex \rightarrow doc/latex/oberdiek/test/iflang-test3.tex
test/iflang-test4.tex \rightarrow doc/latex/oberdiek/test/iflang-test4.tex
test/iflang-test5.tex \rightarrow doc/latex/oberdiek/test/iflang-test5.tex
iflang.dtx \rightarrow source/latex/oberdiek/iflang.dtx
```

If you have a \texttt{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.

\textsuperscript{1}http://ctan.org/pkg/iflang
4.4 Refresh file name databases

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

4.5 Some details for the interested

Unpacking with \LaTeX{}.

The \texttt{.dtx} chooses its action depending on the format:

\begin{itemize}
  \item \texttt{plain \TeX{}}: Run \texttt{docstrip} and extract the files.
  \item \texttt{\LaTeX{}}: Generate the documentation.
\end{itemize}

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

\begin{verbatim}
  \latex \let\install=y\input{iflang.dtx}
\end{verbatim}

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

\textbf{Generating the documentation.}

You can use both the \texttt{.dtx} or the \texttt{.drv} to generate the documentation. The process can be configured by the configuration file \texttt{\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{pdf\LaTeX{}}:

\begin{verbatim}
  pdflatex iflang.dtx
  makeindex -s gind.ist iflang.idx
  pdflatex iflang.dtx
  makeindex -s gind.ist iflang.idx
  pdflatex iflang.dtx
\end{verbatim}

5 Catalogue

The following XML file can be used as source for the \TeX{} Catalogue. The elements \texttt{caption} and \texttt{description} are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is \texttt{iflang.xml}.

\begin{verbatim}
  556 ⟨*catalogue⟩
  557   <?xml version='1.0' encoding='us-ascii'?>
  558   <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
  559   <entry datestamp='$Date$' modifier='$Author$' id='iflang'>
  560     <name>iflang</name>
  561     <caption>Expandable checks for the current language.</caption>
  562     <authorref id='auth:oberdiek'/>
  563     <copyright owner='Heiko Oberdiek' year='2007'/>
  564     <license type='lppl1.3'/>
  565     <version number='1.7'/>
  566     <description>
  567       This package provides expandable checks for the current language
  568       based on macro \texttt{\textbackslash languagename} or hyphenation patterns.
  569       \textless p/\textgreater
  570       The package is part of the \texttt{xref refid='oberdiek'>oberdiek</xref> bundle.
  571     </description>
  572     <documentation details='Package documentation'
  573       href='ctan:/macros/latex/contrib/oberdiek/iflang.pdf'/>
  574     <ctan file='true' path='\texttt{ctan}/macros/latex/contrib/oberdiek/iflang.dtx'/>
  575     <miktex location='oberdiek'/>
  576     <texlive location='oberdiek'/>
  577     <install path='\texttt{ctan}/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
  578   </entry>
  579 ⟨/catalogue⟩
\end{verbatim}
6 Acknowledgement

I wish to thank:

Markus Kohm Useful hints for version 1.2.

7 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 \rightarrow \IfLangStrEqual (Gabriele Balducci).
- Catcode section rewritten.

[2007/11/11 v1.5]
- Use of package pdftexcmds for Lua\TeX{} support.

[2016/05/16 v1.6]
- Documentation updates.

[2018/01/21 v1.7]
- Fix test for etex.src.

8 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.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>@firstoftwo</th>
<th>111</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>286</td>
<td>119, 128, 143, 206, 213, 254, 264</td>
</tr>
<tr>
<td>%</td>
<td>362</td>
<td>@gobble 292, 300</td>
</tr>
<tr>
<td>@</td>
<td>287, 360</td>
<td>@onelevel@sanitize, 423, 514, 530, 532</td>
</tr>
<tr>
<td>@PackageNameNoLine</td>
<td>159, 164, 194</td>
<td>@secondoftwo 114</td>
</tr>
<tr>
<td>@PackageWarningNoLine</td>
<td>177, 184, 274</td>
<td>121, 126, 145, 208, 215, 256, 266</td>
</tr>
<tr>
<td>@firstofone</td>
<td>295, 298</td>
<td>@undefined 58</td>
</tr>
</tbody>
</table>
S
\selectlanguage ........................ 408
\space ............................... 178,
185, 195, 275, 342, 343, 351, 439
\strip@prefix ........................... 513, 515

T
\Test .................................... 355, 378
\test .................................... 394, 397, 398, 399,
400, 403, 406, 407, 409, 410,
411, 414, 415, 418, 419, 424,
425, 430, 431, 434, 435, 436,
437, 438, 439, 442, 443, 444, 445
\TestCompare ............................ 517, 537, 538,
539, 540, 547, 548, 549, 550, 551
\TestDefined .............................. 501, 509, 511
\TestGeneric ............................. 464, 474, 475
\TestName ............................... 475, 480, 481, 488, 489
\TestPatterns ........................... 474, 477, 478, 485, 486
\the ...................................... 377, 78, 79,
80, 81, 82, 83, 84, 97, 322, 342, 343
\Tmp@EnsureCode ........................ 94, 101, 102,

U
\UNDEFINED .............................. 452, 455, 496, 497, 510
\uselanguage ............................ 327, 333, 342
\usepackage ............................. 387, 390, 391

W
\write ..................................... 23, 52

X
\x ........................................ 12, 14, 15, 17, 19, 20,
22, 26, 28, 51, 56, 66, 75, 87, 466, 468, 519, 527
\x austrian .............................. 428, 430, 433, 436, 441, 444
\x german ............................... 429, 431, 437, 445

Y
\y .......................................... 467, 468