The iflang package

Heiko Oberdiek*  
<heiko.oberdiek at googlemail.com>  
2018/01/21 v1.7

Abstract

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

Contents

1 Documentation 2

2 Implementation 3

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 5

2.2.3 Macros for messages 5

2.2.4 Support for etex.src 5

2.3 \IfLanguagePatterns 6

2.4 \IfLanguageName 6

2.5 Check plausibility of \languagename 8

3 Test 8

3.1 Catcode checks for loading 8

3.2 Test with \LaTeX{} 10

3.3 Test with plain \TeX{} and \TeX{} 11

3.4 Test with plain \TeX{} and without \TeX/pdf\TeX{} 12

4 Installation 13

4.1 Download 13

4.2 Bundle installation 13

4.3 Package installation 13

4.4 Refresh file name databases 14

4.5 Some details for the interested 14

5 Catalogue 14

6 Acknowledgement 15

7 History 15

[2007/04/10 v1.0] 15
[2007/04/11 v1.1] 15
[2007/04/12 v1.2] 15
[2007/04/26 v1.3] 15
[2007/11/11 v1.5] 15

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

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

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

\begin{verbatim}
\texttt{\textbackslash ifLanguageName} \{\langle lang\rangle\} \{\langle then\rangle\} \{\langle else\rangle\}
\end{verbatim}

Makro \texttt{\textbackslash ifLanguageName} compares language \texttt{\langle lang\rangle} with the current setting of macro \texttt{\textbackslash language}. If both contains the same name then the \texttt{\langle then\rangle} part is called, otherwise the \texttt{\langle else\rangle} part.

The macro is expandable. Thus it can be safely used inside \texttt{\textbackslashedef} or \texttt{\textbackslashcsname}. If case of errors like an undefined \texttt{\textbackslash language} the \texttt{\langle else\rangle} part is executed.

Note: Macro \texttt{\textbackslash ifLanguageName} relies on the fact, that \texttt{\textbackslash language} is set correctly:

Package babel:
Full support of \texttt{\textbackslash language} 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 \texttt{\textbackslash language} to the last language in language.dat, but \texttt{\textbackslash language} (current patterns) is zero and points to the first language. Thus the value of \texttt{\textbackslash language} is basically garbage. Package iflang warns if \texttt{\textbackslash language} and \texttt{\textbackslash language} do not fit. This can be fixed by loading package babel previously.

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

\begin{verbatim}
\texttt{\textbackslash ifLanguagePatterns} \{\langle lang\rangle\} \{\langle then\rangle\} \{\langle else\rangle\}
\end{verbatim}

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

The following naming convention for the pattern are supported:

\begin{verbatim}
babel/language.dat : \texttt{\textbackslash l@\langle language\rangle}
etex.src/language.def : \texttt{\textbackslash lang@\langle language\rangle}
\end{verbatim}

Package iflang looks for \texttt{\textbackslash et@xpatterns} (defined in 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{}:

\begin{verbatim}
\begin{group}
catcode61=5 \catcode48=10 \relax
\catcode13=13 \endlinechar=13
\catcode35=6 \catcode39=12 \catcode44=12 \catcode45=12 \catcode46=12 \catcode58=12 \catcode123=1 \catcode125=2
\expandafter\ifx\csname ver@iflang.sty\endcsname\relax
\else
\def\empty{}\fi
\ifx\empty\else
\expandafter\ifx\csname PackageInfo\endcsname\relax
\x{iflang}{The package is already loaded}
\aftergroup\endinput
\else
\begin{group}
catcode61=5 \catcode48=10 \relax
\catcode13=13 \endlinechar=13
\catcode35=6 \catcode39=12 \catcode44=12 \catcode45=12 \catcode46=12 \catcode47=12 \catcode58=12 \catcode64=11 \catcode91=12 \catcode93=12 \catcode123=1 \catcode125=2
\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
\end{group}
\end{verbatim}

Package identification:

\begin{verbatim}
\begin{group}
catcode61=5 \catcode48=10 \relax
\catcode13=13 \endlinechar=13
\catcode35=6 \catcode39=12 \catcode44=12 \catcode45=12 \catcode46=12 \catcode47=12 \catcode58=12 \catcode64=11 \catcode91=12 \catcode93=12 \catcode123=1 \catcode125=2
\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[#4]{\endgroup
\immediate\write-1{Package: #1\{#2, stopped\}}}\fi
\end{group}
\end{verbatim}
2.2 Tools

2.2.1 Provide some basic macros of \LaTeX

\@firstoftwo
2.2.2 Expandible existence check for macros

\IfLang@IfDefined

2.2.3 Macros for messages

\IfLang@prefix

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 USenglish. Thus, if \languagename is not defined, we can quite safely assume USenglish. As additional safety precaution the actual used patterns are checked.

2.3 \IfLanguagePatterns

2.4 \IfLanguageName
We do not have \texttt{pdf@strcmp} (and \texttt{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 Bacho\TeX{} 2005: \url{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 \texttt{language name} might consists of further macros, we need a variant that allows macros in the first string, too.

\begin{verbatim}
\def\IfLang@StrNil{\relax}\%  
\def\IfLang@StrEqual#1{\%  
  \number\IfLang@StrEqualStart{}{}#1\IfLang@StrNil\%  
}  
\def\IfLang@StrEqualStart#1#2#3{\%  
  \ifx#3\IfLang@StrNil\IfLang@StrEqualStop\%  
  \fi\%  
  \ifcat\noexpand#3\relax\IfLang@StrExpand{#1}{#2}\%  
  \fi\%  
  \IfLang@StrEqualStart{\if#3#1}{#2\fi}  
}\%  
\def\IfLang@StrEqualStop\%  
\def\IfLang@StrExpand#1#2#3\%  
\def\IfLang@@StrExpand#1#2#3\IfLang@StrNil{\%  
  \expandafter\IfLang@@@StrExpand#3\IfLang@StrNil{#1}{#2}\%  
}\%  
\def\IfLang@@@StrExpand#1\IfLang@StrNil#2#3{\%  
  \IfLang@StrEqualStart{#2}{#3}#1\IfLang@StrNil\%  
}\%  
\IfLanguageName\%  
\def\IfLanguageName#1{\%  
  \ifnum\IfLang@IfDefined{language name}{%  
    \if\expandafter\IfLang@StrEqual\expandafter  
    \texttt{language name}{#1}  
    0\%  
    \else\%  
      1\%  
    \fi\%  
    \else\%  
      \IfLanguageName\%  
    \fi\%  
  }{}\%  
}\%  
\input{expandafter}expandafter\endinput
\end{verbatim}
2.5 Check plausibility of \texttt{\languagename}

\begin{verbatim}
\begingroup \expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname languagename\endcsname\relax
\else
\IfLanguagePatterns{\languagename}{\@PackageWarningNoLine{iflang}{Mismatch between \string\language\space\string\languagename}}\fi
\IfLang@AtEnd\langle/\package\rangle
\end{verbatim}

3 Test

3.1 Catcode checks for loading

\begin{verbatim}
\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\ifx\csname loop\endcsname\relax\expandafter\@firstofone\else\expandafter\@gobble\fi{\def\loop#1\repeat{\def\body{#1}\iterate\let\next\iterate\else\let\next\relax\fi}
\def\iterate{\body\let\next\iterate\else\let\next\relax\fi}
\end{verbatim}
\texttt{\next}
\texttt{\let\repeat=\fi}
\texttt{\def\RestoreCatcodes{}}
\texttt{\count@=0 \%}
\texttt{\loop}
\texttt{\edef\RestoreCatcodes{}}
\texttt{\def\RangeCatcodeInvalid#1#2{}}
\texttt{\count@=#1\relax}
\texttt{\loop}
\texttt{\ifnum\count@<#2\relax}
\texttt{\advance\count@ 1 \%}
\texttt{\repeat}
\texttt{\def\space{ }}
\texttt{\expandafter\ifx\csname LoadCommand\endcsname\relax}
\texttt{\def\LoadCommand{\input iflang.sty\relax}}
\texttt{\fi}
\texttt{\def\Test{%}
\texttt{\RangeCatcodeInvalid0{47} \%}
\texttt{\RangeCatcodeInvalid58{64} \%}
\texttt{\RangeCatcodeInvalid91{96} \%}
\texttt{\RangeCatcodeInvalid123{255} \%}
\texttt{\catcode`\@=12 \%}
\texttt{\catcode`\=0 \%}
\texttt{\catcode`\%=14 \%}
\texttt{\catcode`\=	he\catcode\count@ \relax}
\texttt{\catcode`\%=14 \%}
\texttt{\RangeCatcodeCheck0{36}{15} \%}
\texttt{\RangeCatcodeCheck37{37}{14} \%}
\texttt{\RangeCatcodeCheck48{57}{12} \%}
\texttt{\RangeCatcodeCheck58{63}{15} \%}
\texttt{\RangeCatcodeCheck64{64}{12} \%}
\texttt{\RangeCatcodeCheck65{90}{11} \%}
\texttt{\RangeCatcodeCheck91{91}{15} \%}
\texttt{\RangeCatcodeCheck92{92}{0} \%}
\texttt{\RangeCatcodeCheck93{96}{15} \%}
\texttt{\RangeCatcodeCheck97{122}{11} \%}
\newcommand{\RangeCatcodeCheck}[1]{\texttt{(123{15})}\%}
\newcommand{\RestoreCatcodes}{\%}
\newcommand{\\Test}{\%}
\begin{document}
\begin{qstest}{IfLanguagePatterns}{language, pattern}
\def\test#1\associates{\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\associates{\IfLanguageName{#1}{true}{false}}{#2}{\%}
\test{ngerman}{true}\%
\test{naustrian}{false}\%
\selectlanguage{naustrian}\%
\test{ngerman}{false}\%
\test{naustrian}{true}\%
\def\languagename{naustrian}\%
\def\languagename{\string naustrian}\%
\edef\languagename{naustrian}\%
\makeatletter\@onelevel@sanitize\languagename\%
\test{naustrian}{true}\%
\test{ngerman}{false}\%
\def\languagename{naustrian}\%
\def\xaustrian{naustrian}\%
\def\xgerman{ngerman}\%
\test{xaustrian}{naustrian}\%
\test{xgerman}{ngerman}\%
\test{\xaustrian}{true}\%
\test{\xgerman}{false}\%
\def\languagename{\xaustrian}\%
\test{naustrian}{true}\%
\end{qstest}
\end{document}

\NeedsTeXFormat{LaTeX2e}
\documentclass{minimal}
\usepackage{babel}
\usepackage{iflang}
\begin{document}
\begin{qstest}{IfLanguagePatterns}{language, pattern}
\test{ngerman}{true}\%
\test{naustrian}{true}\%
\test{english}{false}\%
\test{foobar}{false}\%
\end{qstest}
\begin{qstest}{IfLanguageName}{language, name}
\test{ngerman}{true}\%
\test{naustrian}{false}\%
\selectlanguage{naustrian}\%
\test{ngerman}{false}\%
\test{naustrian}{true}\%
\def\languagename{naustrian}\%
\def\languagename{\string naustrian}\%
\edef\languagename{naustrian}\%
\makeatletter\@onelevel@sanitize\languagename\%
\test{naustrian}{true}\%
\test{ngerman}{false}\%
\def\languagename{naustrian}\%
\def\xaustrian{naustrian}\%
\def\xgerman{ngerman}\%
\test{xaustrian}{naustrian}\%
\test{xgerman}{ngerman}\%
\test{\xaustrian}{true}\%
\test{\xgerman}{false}\%
\def\languagename{\xaustrian}\%
\test{naustrian}{true}\%
\end{qstest}
\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\UNDEFINED true\else false\fi}{true}\
\end{qstest}
\end{document}

3.3 Test with plain \textsc{TeX} and \texttt{ε-TeX}

\begin{verbatim}
\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
\end{verbatim}
3.4 Test with plain \TeX and without $\varepsilon$-\TeX/pd\TeX

\begin{verbatim}
\input iflang.sty
\catcode64=11
\let\TestDefined\UNDEFINED
\let\pdfstrcmp\UNDEFINED
\input iflang.sty
\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>{\edef#1{\expandafter\strip@prefix\meaning#1}}
\def\@onelevel@sanitize#1{\edef#1{\expandafter\strip@prefix\meaning#1}}
\def\TestCompare#1#2#3{\begingroup\edef\x{\if\IfLang@StrEqual{#1}{#2}true\elsefalse\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{\gobble a}{\exch x\b}{}{true}\TestCompare{\exch yz\b}{\exch yz\b}{true}\TestCompare{\c}{\c}{true}\TestCompare{\d}{\d}{true}\csname @@end\endcsname
\end{verbatim}

\end{verbatim}

12
4 Installation

4.1 Download

Package. This package is available on CTAN:\footnote{http://ctan.org/pkg/iflang}

\begin{itemize}
\item [CTAN:macros/latex/contrib/oberdiek/iflang.dtx] The source file.
\end{itemize}

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.

\begin{itemize}
\item [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 \texttt{texmf} in their name are usually organized this way.
\end{itemize}

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):

\begin{verbatim}
unzip oberdiek.tds.zip -d ~/texmf
\end{verbatim}

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):

\begin{verbatim}
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
\end{verbatim}

4.3 Package installation

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

\begin{verbatim}
tex iflang.dtx
\end{verbatim}

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

\begin{itemize}
\item [iflang.sty] \rightarrow tex/generic/oberdiek/iflang.sty
\item [iflang.pdf] \rightarrow doc/latex/oberdiek/iflang.pdf
\item [test/iflang-test1.tex] \rightarrow doc/latex/oberdiek/test/iflang-test1.tex
\item [test/iflang-test2.tex] \rightarrow doc/latex/oberdiek/test/iflang-test2.tex
\item [test/iflang-test3.tex] \rightarrow doc/latex/oberdiek/test/iflang-test3.tex
\item [test/iflang-test4.tex] \rightarrow doc/latex/oberdiek/test/iflang-test4.tex
\item [test/iflang-test5.tex] \rightarrow doc/latex/oberdiek/test/iflang-test5.tex
\item [iflang.dtx] \rightarrow source/latex/oberdiek/iflang.dtx
\end{itemize}

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.
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 \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.

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{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 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}.

```
<catalogue>
  <entry datestamp='$Date$' modifier='$Author$' id='iflang'>
    <name>iflang</name>
    <caption>Expandable checks for the current language.</caption>
    <authorref id='auth:oberdiek'/>
    <copyright owner='Heiko Oberdiek' year='2007'/>
    <license type='lppl1.3'/>
    <version number='1.7'/>
    <description>
      This package provides expandable checks for the current language based on macro \texttt{\languagename} or hyphenation patterns.
      \par
      The package is part of the \texttt{oberdiek} bundle.
    </description>
    <documentation details='Package documentation' href='ctan:/macros/latex/contrib/oberdiek/iflang.pdf'/>
    <ctan file='true' path='macros/latex/contrib/oberdiek/iflang.dtx'/>
    <miktex location='oberdiek'/>
    <texlive location='oberdiek'/>
    <install path='macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
  </entry>
</catalogue>
```

14
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 \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.

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></td>
</tr>
<tr>
<td>%</td>
<td>362</td>
<td></td>
</tr>
<tr>
<td>@</td>
<td>287, 360</td>
<td></td>
</tr>
<tr>
<td>@PackageInfoNoLine</td>
<td>159, 164, 194</td>
<td></td>
</tr>
<tr>
<td>@PackageWarningNoLine</td>
<td>177, 184, 274</td>
<td></td>
</tr>
<tr>
<td>@firstofone</td>
<td>295, 298</td>
<td></td>
</tr>
<tr>
<td>@undefined</td>
<td>58</td>
<td></td>
</tr>
</tbody>
</table>