The hyphsubst package

Heiko Oberdiek*

2016/05/16 v0.3

Abstract

A TEX format file may include alternative hyphenation patterns for a language with a different name. If the naming convention follows babel's rules, then the hyphenation patterns for a language can be replaced by the alternative hyphenation patterns, provided in the format file.

Contents

1 Documentation

1.1 In short

The package is an experimental package that allows the substitution of hyphenation patterns, example:

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

```
\RequirePackage[ngerman=ngerman-x-20080601]{hyphsubst}
\documentclass{article}
\usepackage[ngerman]{babel}
```

The patterns `ngerman` are replaced by the patterns `ngerman-x-20080601`. The format must contain these patterns and should use the naming scheme of either babel's `language.dat` or `etex.src`'s `language.def`.

1.2 Longer version

Assume the format may contain the following hyphenation patterns (excerpt from `language.dat`):

```
...
ngerman dehyphn.tex
ngerman-x-20071231 dehyphn-x-20071231
ngerman-x-20080601 dehyphn-x-20080601
=ngerman-x-latest % alias for ngerman-x-20080601
...
```

The patterns that contain `-x-` are experimental new patterns for `ngerman`. However, package babel does not provide the use of patterns that do not have the same name as the used language (dialect). The babel system remembers patterns in macros: `\l@`⟨*name*⟩. ε-TEX's `etex.src` uses `\lang@`⟨*name*⟩ instead. In the following we use `babel`'s naming scheme, but `etex.src`'s naming scheme is supported, too.

This package hyphsubst solves the problem by redefining the macro `\l@`⟨*name*⟩ to use other patterns.

`\HyphSubstLet` {⟨*nameA*⟩} {⟨*nameB*⟩}

`\l@`⟨*nameA*⟩ now has the same meaning as `\l@`⟨*nameB*⟩. The patterns for `nameB` must exist. If the patterns for `nameA` exist, then they will be overwritten to use the patterns for `nameB`. Example:

```
\documentclass{article}
\usepackage{hyphsubst}
\HyphSubstLet{ngerman}{ngerman-x-20080601}
\usepackage[ngerman]{babel}
```

Now the patterns `ngerman-x-20080601` are be used.

Or if you want to compare hyphenations:

```
\documentclass{article}
\usepackage{hyphsubst}
  % save original patterns for ngerman in ngerman-saved
\HyphSubstLet{ngerman-saved}{ngerman}
\usepackage[ngerman]{babel}
\begin{document}
  We start with the original patterns for ngerman.
  \HyphSubstLet{ngerman}{ngerman-x-latest}%
  Now we are using ngerman-x-latest.
  \HyphSubstLet{ngerman}{ngerman-saved}%
  Again we are using the original patterns.
\end{document}
```

`\HyphSubstIfExists` {⟨*name*⟩} {⟨*then*⟩} {⟨*else*⟩}

Tests if patterns with name ⟨*name*⟩ exist and execute ⟨*then*⟩ in case of success and ⟨*else*⟩ otherwise.

1.3 LaTeX

The package can also be loaded before `\documentclass`:

```
\RequirePackage[ngerman=ngerman-x-20080601]{hyphsubst}
\documentclass{article}
...
```

This allows to put the package in a format file.

Package options are interpreted as 'let' assignments and passed to macro `\HyphSubstLet`:

```
\usepackage[ngerman=ngerman-x-20080601]{hyphsubst}
```

The part before the equal sign is the first argument for `\HyphSubstLet` and the part after the equal sign forms the second argument:

```
\HyphSubstLet{ngerman}{ngerman-x-20080601}
```

Note, this only works for direct package options. Global options are ignored.

1.4 plain TeX

The package can be loaded and used with plain TeX, e.g.:

```
\input hyphsubst.sty
\HyphSubstLet{ngerman}{ngerman-x-latest}
```

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@hyphsubst.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
```

```
      \expandafter\ifx\csname PackageInfo\endcsname\relax
        \def\x#1#2{%
          \immediate\write-1{Package #1 Info: #2.}%
        }%
      \else
        \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
      \fi
      \x{hyphsubst}{The package is already loaded}%
      \aftergroup\endinput
    \fi
  \fi
\endgroup%
```

Package identification:

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % '
  \catcode40=12 % (
  \catcode41=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
      #2[{#3}]%
      \ifx#1\@undefined
        \xdef#1{#3}%
      \fi
      \ifx#1\relax
        \xdef#1{#3}%
      \fi
    }%
  \fi
\expandafter\x\csname ver@hyphsubst.sty\endcsname
\ProvidesPackage{hyphsubst}%
  [2016/05/16 v0.3 Substitute hyphenation patterns (HO)]%
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname HyphSubst@AtEnd\endcsname{%
```

```
      \endlinechar=\the\endlinechar\relax
      \catcode13=\the\catcode13\relax
      \catcode32=\the\catcode32\relax
      \catcode35=\the\catcode35\relax
      \catcode61=\the\catcode61\relax
      \catcode64=\the\catcode64\relax
      \catcode123=\the\catcode123\relax
      \catcode125=\the\catcode125\relax
    }%
  }%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\HyphSubst@AtEnd{%
    \HyphSubst@AtEnd
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{39}{12}% '
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{58}{12}% :
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{96}{12}% `
\edef\HyphSubst@AtEnd{\HyphSubst@AtEnd\noexpand\endinput}
```

2.2 Package

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input infwarerr.sty\relax
\else
  \RequirePackage{infwarerr}[2007/09/09]%
\fi
```

\HyphSubst@l

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname et@xlang\endcsname\relax
  \def\HyphSubst@l{l@}%
\else
  \def\HyphSubst@l{lang@}%
\fi
```

\HyphSubstLet

```
\def\HyphSubstLet#1#2{%
  \begingroup
    \def\x{}%
    \expandafter\ifx\csname\HyphSubst@l#2\endcsname\relax
      \@PackageError{hyphsubst}{Unknown pattern `#2'}\@ehc
    \else
      \def\lmsg{}%
```

```
      \expandafter\ifx\csname\HyphSubst@l#1\endcsname\relax
        \edef\msg{%
          New: \expandafter\string\csname\HyphSubst@l#1\endcsname
          \noexpand\MessageBreak
        }%
      \else
        \edef\msg{%
          Redefined: \expandafter\string\csname\HyphSubst@l#1\endcsname
          \noexpand\MessageBreak
          old value: \number\csname\HyphSubst@l#1\endcsname
          \noexpand\MessageBreak
        }%
        \ifnum\csname\HyphSubst@l#1\endcsname=\language
          \edef\x{%
            \noexpand\language=%
                \number\csname\HyphSubst@l#2\endcsname\relax
          }%
          \edef\lmsg{%
            \noexpand\MessageBreak
            \string\language\noexpand\space updated%
          }%
        \fi
      \fi
      \expandafter\global\expandafter\let
          \csname\HyphSubst@l#1\expandafter\endcsname
          \csname\HyphSubst@l#2\endcsname
      \@PackageInfo{hyphsubst}{%
        \msg
        new value: \number\csname\HyphSubst@l#1\endcsname
        \lmsg
      }%
    \fi
  \expandafter\endgroup\x
}
```

\HyphSubstIfExists

```
\def\HyphSubstIfExists#1{%
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname\HyphSubst@l#1\endcsname\relax
    \expandafter\@secondoftwo
  \else
    \expandafter\@firstoftwo
  \fi
}
```

\@firstoftwo

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

\@secondoftwo

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

\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname documentclass\endcsname\relax
  \expandafter\HyphSubst@AtEnd
\fi%
```

```
\DeclareOption*{%
  \expandafter\HyphSubst@Option\CurrentOption==\relax
}
\def\HyphSubst@Option#1=#2=#3\relax{%
  \HyphSubstLet{#1}{#2}%
}
\ProcessOptions*\relax

\HyphSubst@AtEnd%
⟨/package⟩
```

3 Installation

3.1 Download

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

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

CTAN:macros/latex/contrib/oberdiek/hyphsubst.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 hyphsubst.dtx
```

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

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

[1] CTAN:pkg/hyphsubst

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{hyphsubst.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 hyphsubst.dtx
makeindex -s gind.ist hyphsubst.idx
pdflatex hyphsubst.dtx
makeindex -s gind.ist hyphsubst.idx
pdflatex hyphsubst.dtx
```

4 History

[2008/06/07 v0.1]

- First public version.

[2008/06/09 v0.2]

- Support for ε-TEX's `language.def` added.
- Fix for undefined `\lmsg`.

[2016/05/16 v0.3]

- Documentation updates.

5 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.

Symbols