The protecteddef package

Heiko Oberdiek
<heiko.oberdiek at googlemail.com>

2011/01/31 v1.0

Abstract

This packages provides `\ProtectedDef` for defining robust macros for both plain TEX and LATEX. First ε-TEX's `\protected` is tried, then LATEX's `\DeclareRobustCommand` is used. Otherwise the macro is not made robust.

Contents

1 Documentation

Many of my packages work for both formats plain TEX and LATEX, even iniTEX is often supported. It would be nice if fragile macros could be protected and made robust. However the different format worlds offer different solutions.

1.1 The LaTeX's way

Usually `\newcommand` is used to define macros. It provides a check if the command to be defined is already defined or cannot be defined for other reasons.

For making robust macros LaTeX provides `\DeclareRobustCommand`. It shares the syntax with `\newcommand`. However it does not provide latters check. Internally the check is available via `\@ifdefinable`.

Internally the robust macro is using `\protect` with a nested macro definition. The `\protect` infrastructure is a feature of `\LaTeX` and usually not available in other formats.

1.2 The ε-TeX's way

The need for robust macros is addressed in `\eTeX`. It provides `\protected` that modifies the behaviour of `\def` in a similar way as `\long`. A protected macro does not expand in some expandable contexts like writing to a file or `\edef`.

1.3 The way of this package

The package tries to find the available protection mechanism. First it looks for `\eTeX`'s `\protected`, then it uses LaTeX's `\DeclareRobustCommand`. If both fails, then the macro remains unprotected.

Additionally, `\LaTeX`'s check, if a macro is already defined is added in all cases. First LaTeX's `\@ifdefinable` is tried to be compatible with LaTeX. If `\@ifdefinable` is not available, then the test is implemented by asserting that the macro is undefined or has the meaning of `\relax`. If the test fails, then in all cases the macro is not defined and an error is thrown.

1.4 Usage

`\ProtectedDef *` {⟨*cmd*⟩} [⟨*num*⟩] {⟨*definition text*⟩}

Macro `\ProtectedDef` follows the syntax of LaTeX's `\newcommand` with the exception that an optional argument is not supported. Macro ⟨*cmd*⟩ is to be defined as `\long` macro without star with ⟨*num*⟩ arguments.

The number of arguments ⟨*num*⟩ must be given as explicite digit `0` upto `9`. Otherwise the part between the argument ⟨*cmd*⟩ and the ⟨*definition text*⟩ is taken as parameter text in the syntax of vanilla TeX. Examples (with `\protected`):

```
\ProtectedDef*{\foo}[1]{\message{#1}}
⇒ \protected\def\foo#1{\message#1}}
```

```
\ProtectedDef\foo{abc}
⇒ \protected\def\foo{abc}
```

```
\ProtectedDef*\foo(#1)<#2>{#1/#2}
⇒ \protected\def\foo(#1)<#2>{#1/#2}
```

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 % '
```

```
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\let\expandafter\x\csname ver@protecteddef.sty\endcsname
  \ifx\x\relax % plain-TeX, first loading
  \else
    \def\empty{}%
    \ifx\x\empty % LaTeX, first loading,
      % variable is initialized, but \ProvidesPackage not yet seen
    \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{protecteddef}{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@protecteddef.sty\endcsname
\ProvidesPackage{protecteddef}%
```

```
  [2011/01/31 v1.0 Definition of protected macros (HO)]%
```

2.2 Catcodes

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname ProDef@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\ProDef@AtEnd{%
    \ProDef@AtEnd
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{38}{4}% &
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{42}{12}% *
\TMP@EnsureCode{45}{12}% -
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{96}{12}% `
\edef\ProDef@AtEnd{\ProDef@AtEnd\noexpand\endinput}
```

2.3 Resources

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \def\TMP@RequirePackage#1[#2]{%
    \begingroup\expandafter\expandafter\expandafter\endgroup
    \expandafter\ifx\csname ver@#1.sty\endcsname\relax
      \input #1.sty\relax
    \fi
  }%
\else
  \let\TMP@RequirePackage\RequirePackage
\fi
\TMP@RequirePackage{ltxcmds}[2010/12/12]%
\TMP@RequirePackage{infwarerr}[2010/04/08]%
```

```
\def\ProDef@temp#1{%
  \expandafter\def\csname ProDef@param[#1]\endcsname % hash-ok
}
\expandafter\def\csname ProDef@param\endcsname{}
\ProDef@temp0{}
\ProDef@temp1{##1}
\ProDef@temp2{##1##2}
\ProDef@temp3{##1##2##3}
\ProDef@temp4{##1##2##3##4}
\ProDef@temp5{##1##2##3##4##5}
\ProDef@temp6{##1##2##3##4##5##6}
\ProDef@temp7{##1##2##3##4##5##7}
\ProDef@temp8{##1##2##3##4##5##7##8}
\ProDef@temp9{##1##2##3##4##5##7##8##9}
```

\ProDef@IfDefinable

```
\ltx@IfUndefined{@ifdefinable}{%
  \long\def\ProDef@IfDefinable#1{%
    \begingroup
      \escapechar=-1 %
    \ltx@ifundefined{\string#1}{%
      \endgroup
      \ltx@firstofone
    }{%
      \expandafter\endgroup
      \expandafter
      \edef\expandafter\ProDef@temp\expandafter{\string#1 }%
      \@PackageError{protecteddef}{%
        Command \ltx@backslashchar\ProDef@temp already defined%
      }\@ehc
      \ltx@gobbletwo
    }%
  }%
}{%
  \long\def\ProDef@IfDefinable#1{%
    \let\ProDef@next\ltx@gobbletwo
    \@ifdefinable{#1}{%
      \let\ProDef@next\ltx@firstofone
    }%
    \ProDef@next
  }%
}

\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname protected\endcsname\relax
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname DeclareRobustCommand\endcsname\relax
    \catcode`\&=14 % comment
  \else
    \newcommand*{\ProtectedDef}{%
      \ltx@ifnextchar*{%
        \ProDef@ProtectedDef
      }{%
        \ProDef@ProtectedDef{}%
      }%
    }%
    \long\def\ProDef@ProtectedDef#1#2#3#{%
      \ProDef@IfDefinable{#2}{%
        \ltx@IfUndefined{ProDef@param#3}{%
          \DeclareRobustCommand*{#2}{}%
          \begingroup
            \escapechar=-1 %
            \def\ProDef@temp{#1}%
```

```
          \edef\x{\endgroup
            \ifx\ProDef@temp\ltx@empty
              \noexpand\long
            \fi
            \noexpand\def
            \expandafter\noexpand\csname\string#2 \endcsname
          }%
          \x#3%
        }{%
          \DeclareRobustCommand#1{#2}#3%
        }%
      }%
    }%
    \expandafter\expandafter\expandafter\ProDef@AtEnd
  \fi
\else
  \catcode`\&=9 % ignore
\fi%
\ProDef@IfDefinable\ProtectedDef{%
&  \protected
  \def\ProtectedDef%
}{%
  \ltx@ifnextchar*{%
    \let\ProDef@long\ltx@empty
    \expandafter\ProDef@ProtectedDef\ltx@gobble
  }{%
    \let\ProDef@long\long
    \ProDef@ProtectedDef
  }%
}
\long\def\ProDef@ProtectedDef#1#2#{%
  \ProDef@IfDefinable{#1}{%
    \ltx@IfUndefined{ProDef@param#2}{%
&     \protected
      \ProDef@long
      \def#1#2%
    }{%
&     \protected
      \ProDef@long
      \expandafter\expandafter\expandafter\def
      \expandafter\expandafter\expandafter#1%
      \csname ProDef@param#2\endcsname
    }%
  }%
}
\ProDef@AtEnd%
```
⟨/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\ifx\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\count@=15 %
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\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
}
\def\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input protecteddef.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{38}{47}{15}%
  \RangeCatcodeCheck{48}{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}%
  \RangeCatcodeCheck{123}{255}{15}%
  \RestoreCatcodes
}
\Test
\csname @@end\endcsname
\end
```

⟨/test1⟩

3.2 Test without LaTeX and \protected

```
⟨*test2⟩
\errorcontextlines=10000 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input protecteddef.sty\relax
  \catcode`\{=1 %
  \catcode`\}=2 %
  \catcode`\#=6 %
\else
  \RequirePackage{protecteddef}[2011/01/31]%
\fi
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname protected\endcsname\relax
  \let\pdef\def
\else
  \def\pdef{\protected\def}%
\fi
\def\msg#{\immediate\write16}
\countdef\errcount=2 %
\long\def\BeginCheck#1\ProtectedDef#2\EndCheck{%
  \begingroup
    \toks0={\ProtectedDef#2}%
    \msg{<<\the\toks0>>}%
  \endgroup
  \setbox0=\hbox{%
    #1%
    \ProtectedDef#2%
    \check\foo
  }%
  \ifdim\wd0=0pt\relax
  \else
    \errmessage{[Definition] Unwanted spaces?!}%
```

```
  \fi
  \setbox0=\hbox{%
    \def\fooinitial{XYZ}%
    \let\foo\fooinitial
    \errcount=0 %
    \expandafter\def\csname @PackageError\endcsname##1##2##3{%
      \advance\errcount by 1 %
    }%
    \expandafter\def\csname @notdefinable\endcsname{%
      \advance\errcount by 1 %
    }%
    \ProtectedDef#2%
    \ifnum\errcount=1 %
    \else
      \errmessage{1 error expected, but found: \the\errcount}%
    \fi
    \ifx\foo\fooinitial
    \else
      \def\space{ }%
      \errmessage{\string\foo\space is overwritten}%
    \fi
  }%
  \ifdim\wd0=0pt\relax
  \else
    \errmessage{[Error] Unwanted spaces?!}%
  \fi
}
\chardef\DeclareVersion=0 %
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname protected\endcsname\relax
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname DeclareRobustCommand\endcsname\relax
  \else
    \chardef\DeclareVersion=1 %
  \fi
\fi
\ifnum\DeclareVersion=0 %
  \def\check#1{%
    \ifx\cmp#1%
      \msg{* Test passed.}%
    \else
      \msg{}%
      \msg{[\meaning#1]}%
      \msg{[\meaning\cmp]}%
      \errmessage{Test failed!}%
    \fi
  }%
\else
  \def\check#1{%
    \begingroup
      \escapechar=-1 %
    \edef\x{\endgroup
      \def\noexpand\cs/{\string#1}%
    }\x
    \edef\CMP{%
      \noexpand\protect
      \expandafter\noexpand\csname\cs/ \endcsname
    }%
    \ifx\CMP#1%
      \expandafter\ifx\csname\cs/ \endcsname\cmp
        \msg{Test passed.}%
      \else
```

```
        \msg{}%
        \msg{[\expandafter\meaning\csname\cs/ \endcsname]}%
        \msg{[\meaning\cmp]}%
        \errmessage{Test failed!}%
      \fi
    \else
      \msg{}%
      \msg{[\meaning#1]}%
      \msg{[\meaning\CMP]}%
      \errmessage{Test failed!}%
    \fi
  }%
\fi

\tracingmacros=1

\BeginCheck
  \pdef\cmp{}%
  \ProtectedDef*\foo{}%
\EndCheck

\BeginCheck
  \pdef\cmp{}%
  \ProtectedDef*\foo[0]{}%
\EndCheck

\BeginCheck
  \pdef\cmp#1{<#1>}%
  \ProtectedDef*\foo[1]{<#1>}%
\EndCheck

\BeginCheck
  \pdef\cmp(#1){<#1>}%
  \ProtectedDef*\foo(#1){<#1>}%
\EndCheck

\BeginCheck
  \long\pdef\cmp{}%
  \ProtectedDef\foo{}%
\EndCheck

\BeginCheck
  \long\pdef\cmp{}%
  \ProtectedDef\foo[0]{}%
\EndCheck

\BeginCheck
  \long\pdef\cmp#1{<#1>}%
  \ProtectedDef\foo[1]{<#1>}%
\EndCheck

\BeginCheck
  \long\pdef\cmp(#1){<#1>}%
  \ProtectedDef\foo(#1){<#1>}%
\EndCheck

\csname @@end\endcsname\end
```
⟨/test2⟩

4 Installation

4.1 Download

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

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

CTAN:macros/latex/contrib/oberdiek/protecteddef.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: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 TEX:

```
tex protecteddef.dtx
```

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

```
protecteddef.sty            → tex/generic/oberdiek/protecteddef.sty
protecteddef.pdf            → doc/latex/oberdiek/protecteddef.pdf
test/protecteddef-test1.tex → doc/latex/oberdiek/test/protecteddef-test1.tex
test/protecteddef-test2.tex → doc/latex/oberdiek/test/protecteddef-test2.tex
protecteddef.dtx            → source/latex/oberdiek/protecteddef.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.

4.4 Refresh file name databases

If your TEX distribution (teTEX, mikTEX, ...) relies on file name databases, you must refresh these. For example, teTEX users run `texhash` or `mktexlsr`.

[1] ftp://ftp.ctan.org/tex-archive/

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 pdftk, e.g. unpack the file into the current directory:

```
pdftk protecteddef.pdf unpack_files output .
```

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

5 History

[2011/01/31 v1.0]

- First public version.

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.