

The makerobust package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2006/03/18 v1.0

Abstract

Package makerobust provides `\MakeRobustCommand` that converts an existing macro to a robust one.

Contents

1	User interface	1
1.1	Example	2
2	Implementation	2
3	Installation	3
3.1	Download	3
3.2	Bundle installation	3
3.3	Package installation	3
3.4	Refresh file name databases	4
3.5	Some details for the interested	4
4	History	4
	[2006/03/18 v1.0]	4
5	Index	5

1 User interface

L^AT_EX offers `\DeclareRobustCommand` to define a robust macro that does not break if it is used in moving arguments. Sometimes a macro is already defined, but not robust. For example, `\(` and `\)` are not robust, inside `\section` the user must use `\protect` explicitly. This could be avoided by making `\(` and `\)` robust.

`\MakeRobustCommand{⟨cmd⟩}`

`\MakeRobustCommand` redefines the macro `⟨cmd⟩` by using `\DeclareRobustCommand` and the existing definition of the macro `⟨cmd⟩`.

- It is an error if `⟨cmd⟩` is undefined. If you want to define a robust command, then you can use `\DeclareRobustCommand` directly.
- If the macro has previously been defined by `\DeclareRobustCommand` then the redefinition of `\MakeRobustCommand` is omitted, because the macro is already robust. Only an information entry is written to the `.log` file. Thus you do not get a warning or an error if the macro is already robust because of an updated L^AT_EX or package that defines the macro.

- Two macros are defined for a macro, defined by `\DeclareRobustCommand`. Example:

```
\DeclareRobustCommand{\foobar}{definition text}
```

Then the macro “`\foobar`” contains the protection code and, depending on the protection mode, calls the internal macro “`\foobar` ”. Notice the space at the end of the macro name. This internal macro “`\foobar` ” now contains the definition “`definition text`”, given in `\DeclareRobustCommand`.

Sometimes it can happen, that the internal macro already exists. This can be caused by a previous `\DeclareRobustCommand` followed by `\renewcommand`. Then the redefinition by `\MakeRobustCommand` would be safe.

However, it can also be possible that the macro is already robust, using the internal macro, but with a different protection code. The redefinition by `\MakeRobustCommand` would then generate an infinite loop.

Therefore `\MakeRobustCommand` raises an error message, if the internal macro (with space at the end) already exists.

1.1 Example

```
1 \example
2 \documentclass{article}
3 \usepackage{makerobust}
4 \MakeRobustCommand{(
5 \MakeRobustCommand\)
6 \pagestyle{headings}
7 \begin{document}
8 \tableofcontents
9 \section{Einstein:  $E=mc^2$ }
10 \newpage
11 Second page.
12 \end{document}
13 \example
```

2 Implementation

```
14 \package
15 \NeedsTeXFormat{LaTeX2e}
16 \ProvidesPackage{makerobust}%
17 [2006/03/18 v1.0 Making a macro robust (HO)]%
18 \def\MakeRobustCommand#1{%
19   \begingroup
20   \ifundefined{\expandafter\@gobble\string#1}{%
21     \endgroup
22     \PackageError{makerobust}{%
23       Macro \string'\string#1\string' is not defined%
24     }\@ehc
25   }{%
26     \global\let\MR@temp#1%
27     \let#1\@undefined
28     \expandafter\let\expandafter\MR@temp
29       \csname\expandafter\@gobble\string#1\endcsname
30     \DeclareRobustCommand#1{%
31       \ifx#1\MR@temp
32         \endgroup
33         \PackageInfo{makerobust}{%
34           \string'\string#1\string' is already robust%
35         }%
36       \else
37         \ifundefined{MR@temp}{%

```

```

38     \global\let\MR@gtemp#1%
39     \endgroup
40     \expandafter\let\csname\expandafter\@gobble\string#1 \endcsname#1%
41     \let#1\MR@gtemp
42 }{%
43     \endgroup
44     \PackageError{makerobust}{%
45         Internal macro \string'\string#1 \string' already exists%
46     }\@ehc
47 }%
48 \fi
49 }%
50 }
51 \</package>

```

3 Installation

3.1 Download

Package. This package is available on CTAN¹:

[CTAN:macros/latex/contrib/oberdiek/makerobust.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/makerobust.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:macros/latex/contrib/oberdiek/oberdiek-tds.zip](#)

TDS refers to the standard “A Directory Structure for T_EX Files” ([CTAN:tds/tds.pdf](#)). 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
```

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/
```

3.3 Package installation

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

```
tex makerobust.dtx
```

¹<http://ftp.ctan.org/tex-archive/>

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

```
makerobust.sty      → tex/latex/oberdiek/makerobust.sty
makerobust.pdf      → doc/latex/oberdiek/makerobust.pdf
makerobust-example.tex → doc/latex/oberdiek/makerobust-example.tex
makerobust.dtx      → source/latex/oberdiek/makerobust.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`.

3.4 Refresh file name databases

If your \TeX distribution (`te \TeX` , `mik \TeX` , ...) relies on file name databases, you must refresh these. For example, `te \TeX` users run `texhash` or `mktextlsr`.

3.5 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 makerobust.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{makerobust.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 `pdf \LaTeX` :

```
pdflatex makerobust.dtx
makeindex -s gind.ist makerobust.idx
pdflatex makerobust.dtx
makeindex -s gind.ist makerobust.idx
pdflatex makerobust.dtx
```

4 History

[2006/03/18 v1.0]

- First version.

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

Symbols		M	
\(4, 9	\MakeRobustCommand	1, 4, 5, 18
\)	5, 9	\MR@gtemp	26, 31, 38, 41
\@ehc	24, 46	\MR@temp	28
\@gobble	20, 29, 40	N	
\@ifundefined	20, 37	\NeedsTeXFormat	15
\@undefined	27	\newpage	10
B		P	
\begin	7	\PackageError	22, 44
C		\PackageInfo	33
\csname	29, 40	\pagestyle	6
D		\ProvidesPackage	16
\DeclareRobustCommand	30	S	
\documentclass	2	\section	9
E		T	
\end	12	\tableofcontents	8
\endcsname	29, 40	U	
I		\usepackage	3
\ifx	31		