

The `protecteddef` package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2011/01/31 v1.0

Abstract

This package 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	1
1.1	The <code>L^AT_EX</code> 's way	2
1.2	The <code>ε-T_EX</code> 's way	2
1.3	The way of this package	2
1.4	Usage	2
2	Implementation	2
2.1	Reload check and package identification	2
2.2	Catcodes	4
2.3	Resources	4
3	Test	6
3.1	Catcode checks for loading	6
3.2	Test without <code>L^AT_EX</code> and <code>\protected</code>	8
4	Installation	11
4.1	Download	11
4.2	Bundle installation	11
4.3	Package installation	11
4.4	Refresh file name databases	11
4.5	Some details for the interested	12
5	Catalogue	12
6	History	13
	[2011/01/31 v1.0]	13
7	Index	13

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 L^AT_EX'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 L^AT_EX provides `\DeclareRobustCommand`. It shares the syntax with `\newcommand`. However it does not provide letters 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 -T_EX'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 L^AT_EX'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 L^AT_EX's `\@ifdefinable` is tried to be compatible with L^AT_EX. 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 L^AT_EX'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 explicit 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 T_EX. 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 L^AT_EX.

```
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@protecteddef.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
21 \expandafter\ifx\csname PackageInfo\endcsname\relax
22 \def\x#1#2{%
23 \immediate\write-1{Package #1 Info: #2.}%
24 }%
25 \else
26 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
27 \fi
28 \x{protecteddef}{The package is already loaded}%
29 \aftergroup\endinput
30 \fi
31 \fi
32 \endgroup%

```

Package identification:

```

33 \begingroup\catcode61\catcode48\catcode32=10\relax%
34 \catcode13=5 % ^~M
35 \endlinechar=13 %
36 \catcode35=6 % #
37 \catcode39=12 % '
38 \catcode40=12 % (
39 \catcode41=12 % )
40 \catcode44=12 % ,
41 \catcode45=12 % -
42 \catcode46=12 % .
43 \catcode47=12 % /
44 \catcode58=12 % :
45 \catcode64=11 % @
46 \catcode91=12 % [
47 \catcode93=12 % ]
48 \catcode123=1 % {
49 \catcode125=2 % }
50 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
51 \def\x#1#2#3[#4]{\endgroup
52 \immediate\write-1{Package: #3 #4}%
53 \xdef#1{#4}%
54 }%
55 \else
56 \def\x#1#2[#3]{\endgroup
57 #2[#3]}%
58 \ifx#1@undefined
59 \xdef#1{#3}%
60 \fi
61 \ifx#1\relax
62 \xdef#1{#3}%
63 \fi
64 }%
65 \fi
66 \expandafter\x\csname ver@protecteddef.sty\endcsname
67 \ProvidesPackage{protecteddef}%

```

2.2 Catcodes

```

69 \begingroup\catcode61\catcode48\catcode32=10\relax%
70 \catcode13=5 % ^~M
71 \endlinechar=13 %
72 \catcode123=1 % {
73 \catcode125=2 % }
74 \catcode64=11 % @
75 \def\x{\endgroup
76   \expandafter\edef\csname ProDef@AtEnd\endcsname{%
77     \endlinechar=\the\endlinechar\relax
78     \catcode13=\the\catcode13\relax
79     \catcode32=\the\catcode32\relax
80     \catcode35=\the\catcode35\relax
81     \catcode61=\the\catcode61\relax
82     \catcode64=\the\catcode64\relax
83     \catcode123=\the\catcode123\relax
84     \catcode125=\the\catcode125\relax
85   }%
86 }%
87 \x\catcode61\catcode48\catcode32=10\relax%
88 \catcode13=5 % ^~M
89 \endlinechar=13 %
90 \catcode35=6 % #
91 \catcode64=11 % @
92 \catcode123=1 % {
93 \catcode125=2 % }
94 \def\TMP@EnsureCode#1#2{%
95   \edef\ProDef@AtEnd{%
96     \ProDef@AtEnd
97     \catcode#1=\the\catcode#1\relax
98   }%
99   \catcode#1=#2\relax
100 }
101 \TMP@EnsureCode{38}{4}% &
102 \TMP@EnsureCode{40}{12}% (
103 \TMP@EnsureCode{41}{12}% )
104 \TMP@EnsureCode{42}{12}% *
105 \TMP@EnsureCode{45}{12}% -
106 \TMP@EnsureCode{46}{12}% .
107 \TMP@EnsureCode{47}{12}% /
108 \TMP@EnsureCode{91}{12}% [
109 \TMP@EnsureCode{93}{12}% ]
110 \TMP@EnsureCode{96}{12}% `
111 \edef\ProDef@AtEnd{\ProDef@AtEnd\noexpand\endinput}

```

2.3 Resources

```

112 \begingroup\expandafter\expandafter\expandafter\endgroup
113 \expandafter\ifx\csname RequirePackage\endcsname\relax
114   \def\TMP@RequirePackage#1[#2]{%
115     \begingroup\expandafter\expandafter\expandafter\endgroup
116     \expandafter\ifx\csname ver@#1.sty\endcsname\relax
117       \input #1.sty\relax
118     \fi
119   }%
120 \else
121   \let\TMP@RequirePackage\RequirePackage
122 \fi
123 \TMP@RequirePackage{ltxcmds}[2010/12/12]%
124 \TMP@RequirePackage{infwarerr}[2010/04/08]%

```

```

125 \def\ProDef@temp#1{%
126   \expandafter\def\csname ProDef@param[#1]\endcsname % hash-ok
127 }
128 \expandafter\def\csname ProDef@param\endcsname{}
129 \ProDef@temp0{}
130 \ProDef@temp1{##1}
131 \ProDef@temp2{##1##2}
132 \ProDef@temp3{##1##2##3}
133 \ProDef@temp4{##1##2##3##4}
134 \ProDef@temp5{##1##2##3##4##5}
135 \ProDef@temp6{##1##2##3##4##5##6}
136 \ProDef@temp7{##1##2##3##4##5##7}
137 \ProDef@temp8{##1##2##3##4##5##7##8}
138 \ProDef@temp9{##1##2##3##4##5##7##8##9}

```

\ProDef@IfDefinable

```

139 \ltx@ifundefined{@ifdefinable}{%
140   \long\def\ProDef@IfDefinable#1{%
141     \begingroup
142     \escapechar=-1 %
143     \ltx@ifundefined{\string#1}{%
144       \endgroup
145       \ltx@firstofone
146     }{%
147       \expandafter\endgroup
148       \expandafter
149       \edef\expandafter\ProDef@temp\expandafter{\string#1 }%
150       \@PackageError{protecteddef}{%
151         Command \ltx@backslashchar\ProDef@temp already defined%
152       }\@ehc
153       \ltx@gobbletwo
154     }%
155   }%
156 }{%
157   \long\def\ProDef@IfDefinable#1{%
158     \let\ProDef@next\ltx@gobbletwo
159     \@ifdefinable{#1}{%
160       \let\ProDef@next\ltx@firstofone
161     }%
162     \ProDef@next
163   }%
164 }

165 \begingroup\expandafter\expandafter\expandafter\endgroup
166 \expandafter\ifx\csname protected\endcsname\relax
167   \begingroup\expandafter\expandafter\expandafter\endgroup
168   \expandafter\ifx\csname DeclareRobustCommand\endcsname\relax
169     \catcode`\&=14 % comment
170   \else
171     \newcommand*{\ProtectedDef}{%
172       \ltx@ifnextchar*{%
173         \ProDef@ProtectedDef
174       }{%
175         \ProDef@ProtectedDef{}%
176       }%
177     }%
178     \long\def\ProDef@ProtectedDef#1#2#3{%
179       \ProDef@IfDefinable{#2}{%
180         \ltx@ifundefined{ProDef@param#3}{%
181           \DeclareRobustCommand*{#2}{}%
182         }%
183         \begingroup
184           \escapechar=-1 %
185           \def\ProDef@temp{#1}%

```

```

185         \edef\x{\endgroup
186         \ifx\ProDef@temp\ltx@empty
187         \noexpand\long
188         \fi
189         \noexpand\def
190         \expandafter\noexpand\csname\string#2 \endcsname
191         }%
192         \x#3%
193     }{%
194         \DeclareRobustCommand#1{#2}#3%
195     }%
196 }%
197 }%
198 \expandafter\expandafter\expandafter\ProDef@AtEnd
199 \fi
200 \else
201 \catcode`\&=9 % ignore
202 \fi%
203 \ProDef@IfDefinable\ProtectedDef{%
204 & \protected
205 \def\ProtectedDef%
206 }{%
207 \ltx@ifnextchar*{%
208 \let\ProDef@long\ltx@empty
209 \expandafter\ProDef@ProtectedDef\ltx@gobble
210 }{%
211 \let\ProDef@long\long
212 \ProDef@ProtectedDef
213 }%
214 }
215 \long\def\ProDef@ProtectedDef#1#2#{%
216 \ProDef@IfDefinable{#1}{%
217 \ltx@ifundefined{ProDef@param#2}{%
218 & \protected
219 \ProDef@long
220 \def#1#2%
221 }{%
222 & \protected
223 \ProDef@long
224 \expandafter\expandafter\expandafter\def
225 \expandafter\expandafter\expandafter#1%
226 \csname ProDef@param#2\endcsname
227 }%
228 }%
229 }
230 \ProDef@AtEnd%
231 </package>

```

3 Test

3.1 Catcode checks for loading

```

232 <*test1>
233 \catcode`\{=1 %
234 \catcode`\}=2 %
235 \catcode`\#=6 %
236 \catcode`\@=11 %
237 \expandafter\ifx\csname count@\endcsname\relax
238 \countdef\count@=255 %
239 \fi
240 \expandafter\ifx\csname @gobble\endcsname\relax

```

```

241 \long\def\@gobble#1{}%
242 \fi
243 \expandafter\ifx\csname @firstofone\endcsname\relax
244 \long\def\@firstofone#1{#1}%
245 \fi
246 \expandafter\ifx\csname loop\endcsname\relax
247 \expandafter\@firstofone
248 \else
249 \expandafter\@gobble
250 \fi
251 {%
252 \def\loop#1\repeat{%
253 \def\body{#1}%
254 \iterate
255 }%
256 \def\iterate{%
257 \body
258 \let\next\iterate
259 \else
260 \let\next\relax
261 \fi
262 \next
263 }%
264 \let\repeat=\fi
265 }%
266 \def\RestoreCatcodes{}
267 \count@=0 %
268 \loop
269 \edef\RestoreCatcodes{%
270 \RestoreCatcodes
271 \catcode\the\count@=\the\catcode\count@\relax
272 }%
273 \ifnum\count@<255 %
274 \advance\count@ 1 %
275 \repeat
276
277 \def\RangeCatcodeInvalid#1#2{%
278 \count@=#1\relax
279 \loop
280 \catcode\count@=15 %
281 \ifnum\count@<#2\relax
282 \advance\count@ 1 %
283 \repeat
284 }
285 \def\RangeCatcodeCheck#1#2#3{%
286 \count@=#1\relax
287 \loop
288 \ifnum#3=\catcode\count@
289 \else
290 \errmessage{%
291 Character \the\count@\space
292 with wrong catcode \the\catcode\count@\space
293 instead of \number#3%
294 }%
295 \fi
296 \ifnum\count@<#2\relax
297 \advance\count@ 1 %
298 \repeat
299 }
300 \def\space{ }
301 \expandafter\ifx\csname LoadCommand\endcsname\relax
302 \def\LoadCommand{\input protecteddef.sty\relax}%

```

```

303 \fi
304 \def\Test{%
305   \RangeCatcodeInvalid{0}{47}%
306   \RangeCatcodeInvalid{58}{64}%
307   \RangeCatcodeInvalid{91}{96}%
308   \RangeCatcodeInvalid{123}{255}%
309   \catcode`\@=12 %
310   \catcode`\=0 %
311   \catcode`\%=14 %
312   \LoadCommand
313   \RangeCatcodeCheck{0}{36}{15}%
314   \RangeCatcodeCheck{37}{37}{14}%
315   \RangeCatcodeCheck{38}{47}{15}%
316   \RangeCatcodeCheck{48}{57}{12}%
317   \RangeCatcodeCheck{58}{63}{15}%
318   \RangeCatcodeCheck{64}{64}{12}%
319   \RangeCatcodeCheck{65}{90}{11}%
320   \RangeCatcodeCheck{91}{91}{15}%
321   \RangeCatcodeCheck{92}{92}{0}%
322   \RangeCatcodeCheck{93}{96}{15}%
323   \RangeCatcodeCheck{97}{122}{11}%
324   \RangeCatcodeCheck{123}{255}{15}%
325   \RestoreCatcodes
326 }
327 \Test
328 \csname @@end\endcsname
329 \end
330 </test1>

```

3.2 Test without L^AT_EX and \protected

```

331 (*test2)
332 \errorcontextlines=10000 %
333 \begingroup\expandafter\expandafter\expandafter\endgroup
334 \expandafter\ifx\csname RequirePackage\endcsname\relax
335   \input protecteddef.sty\relax
336   \catcode`\{=1 %
337   \catcode`\}=2 %
338   \catcode`\#=6 %
339 \else
340   \RequirePackage{protecteddef}[2011/01/31]%
341 \fi
342 \begingroup\expandafter\expandafter\expandafter\endgroup
343 \expandafter\ifx\csname protected\endcsname\relax
344   \let\pdef\def
345 \else
346   \def\pdef{\protected\def}%
347 \fi
348 \def\msg#{\immediate\write16}
349 \countdef\errcount=2 %
350 \long\def\BeginCheck#1\ProtectedDef#2\EndCheck{%
351   \begingroup
352     \toks0={\ProtectedDef#2}%
353     \msg{<<\the\toks0>>}%
354   \endgroup
355   \setbox0=\hbox{%
356     #1%
357     \ProtectedDef#2%
358     \check\foo
359   }%
360   \ifdim\wd0=0pt\relax
361   \else
362     \errmessage{[Definition] Unwanted spaces?!}%

```



```

363 \fi
364 \setbox0=\hbox{%
365   \def\fooinitial{XYZ}%
366   \let\foo\fooinitial
367   \errcount=0 %
368   \expandafter\def\csname @PackageError\endcsname##1##2##3{%
369     \advance\errcount by 1 %
370   }%
371   \expandafter\def\csname @notdefinable\endcsname{%
372     \advance\errcount by 1 %
373   }%
374   \ProtectedDef#2%
375   \ifnum\errcount=1 %
376   \else
377     \errmessage{1 error expected, but found: \the\errcount}%
378   \fi
379   \ifx\foo\fooinitial
380   \else
381     \def\space{ }%
382     \errmessage{\string\foo\space is overwritten}%
383   \fi
384 }%
385 \ifdim\wd0=0pt\relax
386 \else
387   \errmessage{[Error] Unwanted spaces?!}%
388 \fi
389 }
390 \chardef\DeclareVersion=0 %
391 \begingroup\expandafter\expandafter\expandafter\endgroup
392 \expandafter\ifx\csname protected\endcsname\relax
393   \begingroup\expandafter\expandafter\expandafter\endgroup
394   \expandafter\ifx\csname DeclareRobustCommand\endcsname\relax
395   \else
396     \chardef\DeclareVersion=1 %
397   \fi
398 \fi
399 \ifnum\DeclareVersion=0 %
400   \def\check#1{%
401     \ifx\cmp#1%
402       \msg{* Test passed.}%
403     \else
404       \msg{%}%
405       \msg{[\meaning#1]}%
406       \msg{[\meaning\cmp]}%
407       \errmessage{Test failed!}%
408     \fi
409   }%
410 \else
411   \def\check#1{%
412     \begingroup
413       \escapechar=-1 %
414       \edef\x{\endgroup
415         \def\noexpand\cs/{\string#1}%
416       }\x
417       \edef\CMP{%
418         \noexpand\protect
419         \expandafter\noexpand\csname\cs/ \endcsname
420       }%
421       \ifx\CMP#1%
422         \expandafter\ifx\csname\cs/ \endcsname\cmp
423           \msg{Test passed.}%
424         \else

```

```

425     \msg{}%
426     \msg{[\expandafter\meaning\csname\cs/ \endcsname]}%
427     \msg{[\meaning\cmp]}%
428     \errmessage{Test failed!}%
429     \fi
430   \else
431     \msg{}%
432     \msg{[\meaning#1]}%
433     \msg{[\meaning\CMP]}%
434     \errmessage{Test failed!}%
435   \fi
436 }%
437 \fi
438
439 \tracingmacros=1
440
441 \BeginCheck
442   \pdef\cmp{}%
443   \ProtectedDef*\foo{}%
444 \EndCheck
445
446 \BeginCheck
447   \pdef\cmp{}%
448   \ProtectedDef*\foo[0]{}%
449 \EndCheck
450
451 \BeginCheck
452   \pdef\cmp#1{<#1>}%
453   \ProtectedDef*\foo[1]{<#1>}%
454 \EndCheck
455
456 \BeginCheck
457   \pdef\cmp(#1){<#1>}%
458   \ProtectedDef*\foo(#1){<#1>}%
459 \EndCheck
460
461 \BeginCheck
462   \long\pdef\cmp{}%
463   \ProtectedDef\foo{}%
464 \EndCheck
465
466 \BeginCheck
467   \long\pdef\cmp{}%
468   \ProtectedDef\foo[0]{}%
469 \EndCheck
470
471 \BeginCheck
472   \long\pdef\cmp#1{<#1>}%
473   \ProtectedDef\foo[1]{<#1>}%
474 \EndCheck
475
476 \BeginCheck
477   \long\pdef\cmp(#1){<#1>}%
478   \ProtectedDef\foo(#1){<#1>}%
479 \EndCheck
480
481 \csname @@end\endcsname\end
482 </test2>

```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

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

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 L^AT_EX. The `.dtx` chooses its action depending on the format:

plain T_EX: Run `docstrip` and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for `docstrip` (really, `docstrip` does not need L^AT_EX), 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 Catalogue

The following XML file can be used as source for the [T_EX Catalogue](#). The elements `caption` and `description` are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is `protecteddef.xml`.

```
483 <!*catalogue>
484 <?xml version='1.0' encoding='us-ascii'?>
485 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
486 <entry datestamp='$Date$' modifier='$Author$' id='protecteddef'>
487   <name>protecteddef</name>
488   <caption>Define protected commands.</caption>
489   <authorref id='auth:oberdiek' />
490   <copyright owner='Heiko Oberdiek' year='2011' />
491   <license type='lppl1.3' />
492   <version number='1.0' />
493   <description>
494     The package defines a command <tt>\ProtectedDef</tt> that will
495     create LaTeX &#x2018;robust&#x2019; command or an e-TeX
496     &#x2018;protected&#x2019; command as appropriate
497     to its environment.
498   <p />
499   The package is part of the <xref refid='oberdiek'>oberdiek</xref> bundle.
500 </description>
501 <documentation details='Package documentation'
502   href='ctan:/macros/latex/contrib/oberdiek/protecteddef.pdf' />
503 <ctan file='true' path='/macros/latex/contrib/oberdiek/protecteddef.dtx' />
```

```

504 <miktex location='oberdiek' />
505 <texlive location='oberdiek' />
506 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip' />
507 </entry>
508 </catalogue>

```

6 History

[2011/01/31 v1.0]

- First public version.

7 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			
<code>\#</code>	<i>235, 338</i>	<code>\csname</code>	
<code>\%</code>	<i>311</i>	<i>14, 21, 50,</i>	
<code>\&</code>	<i>169, 201</i>	<i>66, 76, 113, 116, 126, 128, 166,</i>	
<code>\@</code>	<i>236, 309</i>	<i>168, 190, 226, 237, 240, 243,</i>	
<code>\@PackageError</code>	<i>150</i>	<i>246, 301, 328, 334, 343, 368,</i>	
<code>\@ehc</code>	<i>152</i>	<i>371, 392, 394, 419, 422, 426, 481</i>	
<code>\@firstofone</code>	<i>244, 247</i>	D	
<code>\@gobble</code>	<i>241, 249</i>	<code>\DeclareRobustCommand</code>	<i>181, 194</i>
<code>\@ifdefinable</code>	<i>159</i>	<code>\DeclareVersion</code>	<i>390, 396, 399</i>
<code>\@undefined</code>	<i>58</i>	E	
<code>\@</code>	<i>310</i>	<code>\empty</code>	<i>17, 18</i>
<code>\{</code>	<i>233, 336</i>	<code>\end</code>	<i>329, 481</i>
<code>\}</code>	<i>234, 337</i>	<code>\EndCheck</code>	<i>350, 444,</i>
A		<i>449, 454, 459, 464, 469, 474, 479</i>	
<code>\advance</code>	<i>274, 282, 297, 369, 372</i>	<code>\endcsname</code>	<i>14, 21, 50,</i>
<code>\aftergroup</code>	<i>29</i>	<i>66, 76, 113, 116, 126, 128, 166,</i>	
B		<i>168, 190, 226, 237, 240, 243,</i>	
<code>\BeginCheck</code>	<i>350, 441,</i>	<i>246, 301, 328, 334, 343, 368,</i>	
<i>446, 451, 456, 461, 466, 471, 476</i>		<i>371, 392, 394, 419, 422, 426, 481</i>	
<code>\body</code>	<i>253, 257</i>	<code>\endinput</code>	<i>29, 111</i>
C		<code>\endlinechar</code>	<i>4, 35, 71, 77, 89</i>
<code>\catcode</code>	<i>2, 3, 5, 6, 7, 8,</i>	<code>\errcount</code> .	<i>349, 367, 369, 372, 375, 377</i>
<i>9, 10, 11, 12, 13, 33, 34, 36, 37,</i>		<code>\errmessage</code>	<i>290,</i>
<i>38, 39, 40, 41, 42, 43, 44, 45, 46,</i>		<i>362, 377, 382, 387, 407, 428, 434</i>	
<i>47, 48, 49, 69, 70, 72, 73, 74, 78,</i>		<code>\errorcontextlines</code>	<i>332</i>
<i>79, 80, 81, 82, 83, 84, 87, 88, 90,</i>		<code>\escapechar</code>	<i>142, 183, 413</i>
<i>91, 92, 93, 97, 99, 169, 201, 233,</i>		F	
<i>234, 235, 236, 271, 280, 288,</i>		<code>\foo</code>	<i>358, 366, 379, 382, 443,</i>
<i>292, 309, 310, 311, 336, 337, 338</i>		<i>448, 453, 458, 463, 468, 473, 478</i>	
<code>\chardef</code>	<i>390, 396</i>	<code>\fooinitial</code>	<i>365, 366, 379</i>
<code>\check</code>	<i>358, 400, 411</i>	H	
<code>\CMP</code>	<i>417, 421, 433</i>	<code>\hbox</code>	<i>355, 364</i>
<code>\cmp</code>	<i>401, 406, 422, 427, 442,</i>	I	
<i>447, 452, 457, 462, 467, 472, 477</i>		<code>\ifdim</code>	<i>360, 385</i>
<code>\count@</code>	<i>238, 267,</i>	<code>\ifnum</code>	<i>273, 281, 288, 296, 375, 399</i>
<i>271, 273, 274, 278, 280, 281,</i>		<code>\ifx</code>	<i>15, 18, 21, 50,</i>
<i>282, 286, 288, 291, 292, 296, 297</i>		<i>58, 61, 113, 116, 166, 168, 186,</i>	
<code>\countdef</code>	<i>238, 349</i>	<i>237, 240, 243, 246, 301, 334,</i>	
<code>\cs</code>	<i>415, 419, 422, 426</i>	<i>343, 379, 392, 394, 401, 421, 422</i>	

<code>\immediate</code>	23, 52, 348	<code>\protect</code>	418
<code>\input</code>	117, 302, 335	<code>\protected</code>	204, 218, 222, 346
<code>\iterate</code>	254, 256, 258	<code>\ProtectedDef</code>	2, 171, 203, 205, 350, 352, 357, 374, 443, 448, 453, 458, 463, 468, 473, 478, 494
L			
<code>\LoadCommand</code>	302, 312	<code>\ProvidesPackage</code>	19, 67
<code>\loop</code>	252, 268, 279, 287	R	
<code>\ltx@backslashchar</code>	151	<code>\RangeCatcodeCheck</code>	
<code>\ltx@empty</code>	186, 208	. 285, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324	
<code>\ltx@firstofone</code>	145, 160	<code>\RangeCatcodeInvalid</code>	
<code>\ltx@gobble</code>	209	. 277, 305, 306, 307, 308	
<code>\ltx@gobbletwo</code>	153, 158	<code>\repeat</code>	252, 264, 275, 283, 298
<code>\ltx@ifnextchar</code>	172, 207	<code>\RequirePackage</code>	121, 340
<code>\ltx@ifundefined</code>	139, 180, 217	<code>\RestoreCatcodes</code> ..	266, 269, 270, 325
<code>\ltx@ifundefined</code>	143	S	
M			
<code>\meaning</code> ..	405, 406, 426, 427, 432, 433	<code>\setbox</code>	355, 364
<code>\msg</code> ..	348, 353, 402, 404, 405, 406, 423, 425, 426, 427, 431, 432, 433	<code>\space</code>	291, 292, 300, 381, 382
N			
<code>\newcommand</code>	171	T	
<code>\next</code>	258, 260, 262	<code>\Test</code>	304, 327
<code>\number</code>	293	<code>\the</code>	77, 78, 79, 80, 81, 82, 83, 84, 97, 271, 291, 292, 353, 377
P			
<code>\PackageInfo</code>	26	<code>\TMP@EnsureCode</code> .	94, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110
<code>\pdef</code>	344, 346, 442, 447, 452, 457, 462, 467, 472, 477	<code>\TMP@RequirePackage</code>	114, 121, 123, 124
<code>\ProDef@AtEnd</code>	95, 96, 111, 198, 230	<code>\toks</code>	352, 353
<code>\ProDef@ifDefinable</code>	139, 179, 203, 216	<code>\tracingmacros</code>	439
<code>\ProDef@long</code>	208, 211, 219, 223	W	
<code>\ProDef@next</code>	158, 160, 162	<code>\wd</code>	360, 385
<code>\ProDef@ProtectedDef</code>		<code>\write</code>	23, 52, 348
.	173, 175, 178, 209, 212, 215	X	
<code>\ProDef@temp</code>	125, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 149, 151, 184, 186	<code>\x</code>	14, 15, 18, 22, 26, 28, 51, 56, 66, 75, 87, 185, 192, 414, 416