

# The `eolgrab` package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2011/01/12 v1.0

## Abstract

This package implements a generic argument grabber to catch an argument that is delimited by the line end.

## Contents

<b>1</b>	<b>Documentation</b>	<b>1</b>
1.1	Examples	2
1.1.1	Small L <sup>A</sup> T <sub>E</sub> X document as example	3
1.1.2	L <sup>A</sup> T <sub>E</sub> X document with environments	4
1.2	Limitations	4
<b>2</b>	<b>Implementation</b>	<b>4</b>
2.1	Reload check and package identification	4
2.2	Catcodes	6
2.3	Resources	6
2.4	Macro <code>\eolgrab</code>	7
<b>3</b>	<b>Test</b>	<b>8</b>
3.1	Catcode checks for loading	8
3.2	Tests for plain T <sub>E</sub> X	9
<b>4</b>	<b>Installation</b>	<b>10</b>
4.1	Download	10
4.2	Bundle installation	10
4.3	Package installation	11
4.4	Refresh file name databases	11
4.5	Some details for the interested	11
<b>5</b>	<b>References</b>	<b>12</b>
<b>6</b>	<b>History</b>	<b>12</b>
	[2011/01/12 v1.0]	12
<b>7</b>	<b>Index</b>	<b>12</b>

## 1 Documentation

The starting point for this package was a feature request of Arno Trautmann in the mailing list `texhax`<sup>1</sup> [1]. A macro `\eolsection` should behave like `\section`, but the argument should be delimited by the line end instead of given in curly braces:

```
\eolsection My Title
```

---

<sup>1</sup>Info page for mailing list `texhax`: <http://tug.org/mailman/listinfo/texhax>

Phil Taylor answered this with an implementation for `\eolsection`. Because this feature could be useful for other macros as well, I answered with an implementation of `\eolgrab` as general solution [3].

Both formats plain  $\text{\TeX}$  and  $\text{\LaTeX}$  are supported by the package, see the example for `\eolsection` below.

`\eolgrab {<code>} <argument> <EOL>}`

Macro `\eolgrab` takes two arguments. The first argument is `<code>`, a classical undelimited  $\text{\TeX}$  macro argument. The second argument is delimited by the line end `<EOL>`. The macro calls `<code>` with `<argument>` as argument in curly braces. Because the catcode of the line end is changed, `\eolgrab` will not work in the argument of other macros. Macro `\eolgrab` is made robust if either  $\varepsilon\text{-TeX}$ 's `\protected` or  $\text{\LaTeX}$ 's `\DeclareRobustCommand` is available.

`\eolgrabopt {<code>} <argument> <EOL>}`

Macro `\eolgrabopt` passes `<argument>` as optional argument to `<code>` if `<argument>` is not empty.

```
\eolgrabopt\item foo
```

becomes to

```
\item[{foo}]
```

The curly argument braces are added to support square brackets inside `<argument>`. If the `<argument>` is empty:

```
\eolgrabopt\item
```

then

```
\item
```

is called without optional argument.

## 1.1 Examples

- The line

```
\eolgrab\section My Title
```

is equivalent to

```
\section{My Title}
```

- The next example uses the star form of `\section`. Then the command to be called consists of two tokens. Therefore the first argument of `\eolgrab` needs curly braces:

```
\eolgrab{\section*}My Title
```

becomes

```
\section*{My Title}
```

- Now  $\text{\LaTeX}$ 's `\PackageError` is used. This macro has three arguments, the package or class name, the message text and the help text. A standard help text of  $\text{\LaTeX}$  is used as given in macro `\@ehc`. The second argument, the message text is used as argument, delimited by line end:

```

\eolgrab{\PackageError{foobar}}}%
Some error message text\MessageBreak%
with several lines
\@ehc

```

In the first two lines of the example, the line end is suppressed by the comment character (percent), thus the argument is delimited by the line end of the third line. The result is:

```

\PackageError{foobar}{Some error message text\MessageBreak
with several lines}\@ehc

```

- The original request for macro `\eolsection`, see above, can be implemented easily with the help of `\eolgrab`. Example for  $\text{\LaTeX}$ :

```

\usepackage{eolgrab}
\newcommand*{\eolsection}{\eolgrab\section}

```

Example for plain  $\text{\TeX}$ :

```

\input eolgrab.sty\relax
\def\eolsection{\eolgrab\section}

```

And a sophisticated variant for  $\text{\LaTeX}$  that also supports the star syntax and the optional argument:

```

1  \<example-sec>
2  \documentclass{article}
3  \usepackage{eolgrab}
4  \makeatletter
5  \newcommand*{\eolsection}{%
6    \ifstar{%
7      \eolgrab{\section*}%
8    }{%
9      \ifnextchar[{%
10         \eoloptsection
11       }{%
12         \eolgrab\section
13       }%
14     }%
15   }
16   \newcommand*{\eoloptsection}[1][]{%
17     \eolgrab{\section[#1]}%
18   }
19   \makeatother
20   \begin{document}
21   \tableofcontents
22   \eolsection Section without star and optional argument
23   \eolsection*Section with star
24   \eolsection[Short section title]Long section title
25   \end{document}
26 \</example-sec>

```

### 1.1.1 Small $\text{\LaTeX}$ document as example

```

27 \<example-ltx>
28 \RequirePackage{eolgrab}
29 \eolgrab\documentclass article
30 \eolgrab\begin document
31 \eolgrab\section Hello World
32 \eolgrab\emph Some text
33 \eolgrab\end document
34 \</example-ltx>

```

### 1.1.2 L<sup>A</sup>T<sub>E</sub>X document with environments

```
35 (*example-env)
36 \documentclass{article}
37 \usepackage{eolgrab}
38 \newcommand*{\Begin}{\eolgrab \begin}
39 \newcommand*{\End }{\eolgrab \end }
40 \newcommand*{\Item }{\eolgrabopt\item }
41 \Begin document
42 \Begin itemize
43 \Item
44 first item
45 \Item
46 second item
47 \End itemize
48 \Begin description
49 \Item foo
50 is the first syllable of foobar.
51 \Item bar
52 is the second syllable of foobar.
53 \End description
54 \End document
55 /example-env)
```

## 1.2 Limitations

Macro `\eolgrab` needs to catch the line end. If T<sub>E</sub>X reads a line, then it throws away the line end characters (carriage return, line feed) and removes spaces at the end of the line. Then it adds the character with the character code that is given by `\endlinechar` at the end of the line. The category code of the inserted character is given by the current value of its `\catcode`. If `\endlinechar` is not a valid character code (especially if it is negative), then no character is added.

In plain T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X the standard settings of the inserted endline character is the character with code 13 (or `^M` in T<sub>E</sub>X notation) with catcode 5 (end of line). That means the inserted end of line character behaves like a space token. For example, it is removed after macro names. Therefore `\eolgrab` changes the catcode.

Therefore `\eolgrab` has some limitations:

- Like other verbatim stuff, the macro `\eolgrab` cannot be used in the argument of other macros. `\eolgrab` want to change the catcode of the end of line character. If this character is read before, because it is processed as argument of another macro, the catcode is already set and is not reassigned later if `\eolgrab` changes the category code for this character code.
- The argument must not contain the end of line character. Otherwise the first end of line character is already taken as delimiter, leaving the rest of the line outside the argument.
- Because `\eolgrab` is probably mostly used in the line with the delimited argument. Therefore changes of `\endlinechar` will not affect the current line.

## 2 Implementation

```
56 (*package)
```

### 2.1 Reload check and package identification

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```
57 \begingroup\catcode61\catcode48\catcode32=10\relax%
58 \catcode13=5 % ^M
```

```

59 \endlinechar=13 %
60 \catcode35=6 % #
61 \catcode39=12 % '
62 \catcode44=12 % ,
63 \catcode45=12 % -
64 \catcode46=12 % .
65 \catcode58=12 % :
66 \catcode64=11 % @
67 \catcode123=1 % {
68 \catcode125=2 % }
69 \expandafter\let\expandafter\x\csname ver@eolgrab.sty\endcsname
70 \ifx\x\relax % plain-TeX, first loading
71 \else
72   \def\empty{}%
73   \ifx\x\empty % LaTeX, first loading,
74     % variable is initialized, but \ProvidesPackage not yet seen
75   \else
76     \expandafter\ifx\csname PackageInfo\endcsname\relax
77       \def\x#1#2{%
78         \immediate\write-1{Package #1 Info: #2.}%
79       }%
80     \else
81       \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
82     \fi
83     \x{eolgrab}{The package is already loaded}%
84   \aftergroup\endinput
85   \fi
86 \fi
87 \endgroup%

```

Package identification:

```

88 \begingroup\catcode61\catcode48\catcode32=10\relax%
89 \catcode13=5 % ^^M
90 \endlinechar=13 %
91 \catcode35=6 % #
92 \catcode39=12 % '
93 \catcode40=12 % (
94 \catcode41=12 % )
95 \catcode44=12 % ,
96 \catcode45=12 % -
97 \catcode46=12 % .
98 \catcode47=12 % /
99 \catcode58=12 % :
100 \catcode64=11 % @
101 \catcode91=12 % [
102 \catcode93=12 % ]
103 \catcode123=1 % {
104 \catcode125=2 % }
105 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
106   \def\x#1#2#3[#4]{\endgroup
107     \immediate\write-1{Package: #3 #4}%
108     \xdef#1{#4}%
109   }%
110 \else
111   \def\x#1#2[#3]{\endgroup
112     #2[#3]}%
113   \ifx#1\@undefined
114     \xdef#1{#3}%
115   \fi
116   \ifx#1\relax
117     \xdef#1{#3}%
118   \fi
119 }%

```

```

120 \fi
121 \expandafter\x\csname ver@eolgrab.sty\endcsname
122 \ProvidesPackage{eolgrab}%
123 [2011/01/12 v1.0 Grab line end delimited argument (H0)]%

```

## 2.2 Catcodes

```

124 \begingroup\catcode61\catcode48\catcode32=10\relax%
125 \catcode13=5 % ^^M
126 \endlinechar=13 %
127 \catcode123=1 % {
128 \catcode125=2 % }
129 \catcode64=11 % @
130 \def\x{\endgroup
131 \expandafter\edef\csname eolgrab@AtEnd\endcsname{%
132 \endlinechar=\the\endlinechar\relax
133 \catcode13=\the\catcode13\relax
134 \catcode32=\the\catcode32\relax
135 \catcode35=\the\catcode35\relax
136 \catcode61=\the\catcode61\relax
137 \catcode64=\the\catcode64\relax
138 \catcode123=\the\catcode123\relax
139 \catcode125=\the\catcode125\relax
140 }%
141 }%
142 \x\catcode61\catcode48\catcode32=10\relax%
143 \catcode13=5 % ^^M
144 \endlinechar=13 %
145 \catcode35=6 % #
146 \catcode64=11 % @
147 \catcode123=1 % {
148 \catcode125=2 % }
149 \def\TMP@EnsureCode#1#2{%
150 \edef\eolgrab@AtEnd{%
151 \eolgrab@AtEnd
152 \catcode#1=\the\catcode#1\relax
153 }%
154 \catcode#1=#2\relax
155 }
156 \TMP@EnsureCode{40}{12}% (
157 \TMP@EnsureCode{41}{12}% )
158 \TMP@EnsureCode{42}{12}% *
159 \TMP@EnsureCode{46}{12}% .
160 \TMP@EnsureCode{47}{12}% /
161 \TMP@EnsureCode{91}{12}% [
162 \TMP@EnsureCode{93}{12}% ]
163 \TMP@EnsureCode{94}{7}% ^
164 \edef\eolgrab@AtEnd{\eolgrab@AtEnd\noexpand\endinput}

```

## 2.3 Resources

```

165 \begingroup\expandafter\expandafter\expandafter\endgroup
166 \expandafter\ifx\csname RequirePackage\endcsname\relax
167 \input ltxcmds.sty\relax
168 \input infwarerr.sty\relax
169 \else
170 \RequirePackage{ltxcmds}[2010/12/04]%
171 \RequirePackage{infwarerr}[2010/04/08]%
172 \fi

```

\eolgrab@ifdefinable

```

173 \ltx@ifundefined{@ifdefinable}{%
174 \def\eolgrab@ifdefinable#1#2{%

```

```

175 \ltx@ifundefined{#1}{#2}{%
176 \PackageError{eolgrab}{%
177 Command \ltx@backslashchar#1 already defined%
178 }\@ehc
179 }%
180 }%
181 }{%
182 \def\eolgrab@ifdefinable#1{%
183 \expandafter\@ifdefinable\csname#1\endcsname
184 }%
185 }

```

## 2.4 Macro \eolgrab

\eolgrab

```

186 \eolgrab@ifdefinable{eolgrab}{%
187 \ltx@ifundefined{protected}{%
188 \ltx@ifundefined{DeclareRobustCommand}{%
189 \def\eolgrab#1%
190 }{%
191 \newcommand\eolgrab{}%
192 \DeclareRobustCommand*\eolgrab
193 }%
194 }{%
195 \protected\def\eolgrab#1%
196 }{%
197 \begingroup
198 \endlinechar=13 %
199 \catcode13=\ltx@active
200 \eolgrab@{#1}%
201 }%
202 }

```

\eolgrabopt

```

203 \eolgrab@ifdefinable{eolgrabopt}{%
204 \ltx@ifundefined{protected}{%
205 \ltx@ifundefined{DeclareRobustCommand}{%
206 \def\eolgrabopt#1%
207 }{%
208 \newcommand\eolgrabopt{}%
209 \DeclareRobustCommand*\eolgrabopt
210 }%
211 }{%
212 \protected\def\eolgrabopt#1%
213 }{%
214 \begingroup
215 \endlinechar=13 %
216 \catcode13=\ltx@active
217 \eolgrab@opt{#1}%
218 }%
219 }

220 \begingroup
221 \catcode13=\ltx@active %
222 \ltx@firstofone{\endgroup %

```

\eolgrab@

```

223 \def\eolgrab@#1#2~M{%
224 \endgroup %
225 #1{#2}%
226 }%

```

\eolgrab@opt

```
227 \def\eolgrab@opt#1#2^~M{%
228   \endgroup %
229   \ltx@ifempty{#2}{%
230     #1%
231   }{%
232     #1[{#2}]%
233   }%
234 }%

235 }%

236 \eolgrab@AtEnd%
237 </package>
```

## 3 Test

### 3.1 Catcode checks for loading

```
238 <*test1>

239 \catcode'\{=1 %
240 \catcode'\}=2 %
241 \catcode'\#=6 %
242 \catcode'\@=11 %
243 \expandafter\ifx\csname count@\endcsname\relax
244   \countdef\count@=255 %
245 \fi
246 \expandafter\ifx\csname @gobble\endcsname\relax
247   \long\def\@gobble#1{%
248 \fi
249 \expandafter\ifx\csname @firstofone\endcsname\relax
250   \long\def\@firstofone#1{#1}%
251 \fi
252 \expandafter\ifx\csname loop\endcsname\relax
253   \expandafter\@firstofone
254 \else
255   \expandafter\@gobble
256 \fi
257 {%
258   \def\loop#1\repeat{%
259     \def\body{#1}%
260     \iterate
261   }%
262   \def\iterate{%
263     \body
264     \let\next\iterate
265   \else
266     \let\next\relax
267   \fi
268   \next
269 }%
270 \let\repeat=\fi
271 }%
272 \def\RestoreCatcodes{}
273 \count@=0 %
274 \loop
275   \edef\RestoreCatcodes{%
276     \RestoreCatcodes
277     \catcode\the\count@=\the\catcode\count@\relax
278   }%
279 \ifnum\count@<255 %
```



```

280 \advance\count@ 1 %
281 \repeat
282
283 \def\RangeCatcodeInvalid#1#2{%
284 \count@=#1\relax
285 \loop
286 \catcode\count@=15 %
287 \ifnum\count@<#2\relax
288 \advance\count@ 1 %
289 \repeat
290 }
291 \def\RangeCatcodeCheck#1#2#3{%
292 \count@=#1\relax
293 \loop
294 \ifnum#3=\catcode\count@
295 \else
296 \errmessage{%
297 Character \the\count@\space
298 with wrong catcode \the\catcode\count@\space
299 instead of \number#3%
300 }%
301 \fi
302 \ifnum\count@<#2\relax
303 \advance\count@ 1 %
304 \repeat
305 }
306 \def\space{ }
307 \expandafter\ifx\csname LoadCommand\endcsname\relax
308 \def\LoadCommand{\input eolgrab.sty\relax}%
309 \fi
310 \def\Test{%
311 \RangeCatcodeInvalid{0}{47}%
312 \RangeCatcodeInvalid{58}{64}%
313 \RangeCatcodeInvalid{91}{96}%
314 \RangeCatcodeInvalid{123}{255}%
315 \catcode'\@=12 %
316 \catcode'\=0 %
317 \catcode'\%=14 %
318 \LoadCommand
319 \RangeCatcodeCheck{0}{36}{15}%
320 \RangeCatcodeCheck{37}{37}{14}%
321 \RangeCatcodeCheck{38}{47}{15}%
322 \RangeCatcodeCheck{48}{57}{12}%
323 \RangeCatcodeCheck{58}{63}{15}%
324 \RangeCatcodeCheck{64}{64}{12}%
325 \RangeCatcodeCheck{65}{90}{11}%
326 \RangeCatcodeCheck{91}{91}{15}%
327 \RangeCatcodeCheck{92}{92}{0}%
328 \RangeCatcodeCheck{93}{96}{15}%
329 \RangeCatcodeCheck{97}{122}{11}%
330 \RangeCatcodeCheck{123}{255}{15}%
331 \RestoreCatcodes
332 }
333 \Test
334 \csname @@end\endcsname
335 \end
336 </test1>

```

## 3.2 Tests for plain T<sub>E</sub>X

```

337 (*test2)
338 \input eolgrab.sty\relax
339 \catcode'\{=1 %

```

```

340 \catcode'\}=2 %
341 \eolgrab{\immediate\write16}Hello World
342 \def\TestExpected{foo bar}
343 \eolgrab{\def\TestResult}foo bar
344
345 \ifx\TestExpected\TestResult
346   \immediate\write16{* Ok (foo bar)}%
347 \else
348   \errmessage{Test failed (foo bar)}%
349 \fi
350
351 \begingroup
352   \def\TestExpected{foobar}%
353   \endlinechar=-1 %
354   \eolgrab{\def\TestResult}foo
355   bar
356   ^^M
357   \ifx\TestExpected\TestResult
358     \immediate\write16{* Ok (foobar)}%
359   \else
360     \errmessage{Test failed (foobar)}%
361   \fi
362 \endgroup
363
364 \csname @@end\endcsname\end
365 /test2)

```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>2</sup>:

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

[CTAN:macros/latex/contrib/oberdiek/eolgrab.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 T<sub>E</sub>X 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/
```

---

<sup>2</sup>[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)

### 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 eolgrab.dtx
```

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

```
eolgrab.sty          → tex/generic/oberdiek/eolgrab.sty
eolgrab.pdf          → doc/latex/oberdiek/eolgrab.pdf
example/eolgrab-example-ltx.tex → doc/latex/oberdiek/example/eolgrab-example-ltx.tex
example/eolgrab-example-env.tex → doc/latex/oberdiek/example/eolgrab-example-env.tex
example/eolgrab-example-sec.tex → doc/latex/oberdiek/example/eolgrab-example-sec.tex
test/eolgrab-test1.tex → doc/latex/oberdiek/test/eolgrab-test1.tex
test/eolgrab-test2.tex → doc/latex/oberdiek/test/eolgrab-test2.tex
eolgrab.dtx          → source/latex/oberdiek/eolgrab.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`kTEX`, ...) relies on file name databases, you must refresh these. For example, te`TEX` users run `texhash` or `mktextlsr`.

### 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 eolgrab.pdf unpack_files output .
```

**Unpacking with L<sup>A</sup>T<sub>E</sub>X.** The `.dtx` chooses its action depending on the format:

**plain T<sub>E</sub>X:** Run `docstrip` and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for `docstrip` (really, `docstrip` does not need L<sup>A</sup>T<sub>E</sub>X), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{eolgrab.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 pdfL<sup>A</sup>T<sub>E</sub>X:

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

## 5 References

- [1] Arno Trautmann, *[texhax] read argument until EOL*; mailing list `texhax@tug.org`, 2011-01-06;  
<http://tug.org/pipermail/texhax/2011-January/016517.html>.
- [2] Philip Taylor, *Re: [texhax] read argument until EOL*; mailing list `texhax@tug.org`, 2011-01-06;  
<http://tug.org/pipermail/texhax/2011-January/016519.html>.
- [3] Heiko Oberdiek, *Re: [texhax] read argument until EOL*; mailing list `texhax@tug.org`, 2011-01-06;  
<http://tug.org/pipermail/texhax/2011-January/016526.html>.

## 6 History

[2011/01/12 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		239, 240, 241, 242, 277, 286, 294, 298, 315, 316, 317, 339, 340
<code>\#</code> .....	241	
<code>\%</code> .....	317	
<code>\@</code> .....	242, 315	
<code>\@PackageError</code> .....	176	
<code>\@ehc</code> .....	178	
<code>\@firstofone</code> .....	250, 253	
<code>\@gobble</code> .....	247, 255	
<code>\@ifdefinable</code> .....	183	
<code>\@ifnextchar</code> .....	9	
<code>\@ifstar</code> .....	6	
<code>\@undefined</code> .....	113	
<code>\@</code> .....	316	
<code>\{</code> .....	239, 339	
<code>\}</code> .....	240, 340	
A		
<code>\advance</code> .....	280, 288, 303	
<code>\aftergroup</code> .....	84	
B		
<code>\Begin</code> .....	38, 41, 42, 48	
<code>\begin</code> .....	20, 30, 38	
<code>\body</code> .....	259, 263	
C		
<code>\catcode</code> .....	57, 58, 60, 61, 62, 63, 64, 65, 66, 67, 68, 88, 89, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 124, 125, 127, 128, 129, 133, 134, 135, 136, 137, 138, 139, 142, 143, 145, 146, 147, 148, 152, 154, 199, 216, 221,	
<code>\count@</code> .....	244, 273, 277, 279, 280, 284, 286, 287, 288, 292, 294, 297, 298, 302, 303	
<code>\countdef</code> .....	244	
<code>\csname</code> .....	69, 76, 105, 121, 131, 166, 183, 243, 246, 249, 252, 307, 334, 364	
D		
<code>\DeclareRobustCommand</code> .....	192, 209	
<code>\documentclass</code> .....	2, 29, 36	
E		
<code>\emph</code> .....	32	
<code>\empty</code> .....	72, 73	
<code>\End</code> .....	39, 47, 53, 54	
<code>\end</code> .....	25, 33, 39, 335, 364	
<code>\endcsname</code> .....	69, 76, 105, 121, 131, 166, 183, 243, 246, 249, 252, 307, 334, 364	
<code>\endinput</code> .....	84, 164	
<code>\endlinechar</code> .....	59, 90, 126, 132, 144, 198, 215, 353	
<code>\eolgrab</code> .....	2, 7, 12, 17, 29, 30, 31, 32, 33, 38, 39, <u>186</u> , 341, 343, 354	
<code>\eolgrab@</code> .....	200, <u>223</u>	
<code>\eolgrab@AtEnd</code> .....	150, 151, 164, 236	
<code>\eolgrab@ifdefinable</code> ..	<u>173</u> , 186, 203	
<code>\eolgrab@opt</code> .....	217, <u>227</u>	
<code>\eolgrabopt</code> .....	2, 40, <u>203</u>	
<code>\eoloptsection</code> .....	10, 16	
<code>\eolsection</code> .....	5, 22, 23, 24	
<code>\errmessage</code> .....	296, 348, 360	

<b>I</b>		<b>R</b>	
<code>\ifnum</code> .....	279, 287, 294, 302	<code>\RangeCatcodeCheck</code> .....	291, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330
<code>\ifx</code> ..	70, 73, 76, 105, 113, 116, 166, 243, 246, 249, 252, 307, 345, 357	<code>\RangeCatcodeInvalid</code> .....	283, 311, 312, 313, 314
<code>\immediate</code> ....	78, 107, 341, 346, 358	<code>\repeat</code> .....	258, 270, 281, 289, 304
<code>\input</code> .....	167, 168, 308, 338	<code>\RequirePackage</code> .....	28, 170, 171
<code>\Item</code> .....	40, 43, 45, 49, 51	<code>\RestoreCatcodes</code> ..	272, 275, 276, 331
<code>\item</code> .....	40		
<code>\iterate</code> .....	260, 262, 264		
<b>L</b>		<b>S</b>	
<code>\LoadCommand</code> .....	308, 318	<code>\section</code> .....	7, 12, 17, 31
<code>\loop</code> .....	258, 274, 285, 293	<code>\space</code> .....	297, 298, 306
<code>\ltx@active</code> .....	199, 216, 221		
<code>\ltx@backslashchar</code> .....	177	<b>T</b>	
<code>\ltx@firstofone</code> .....	222	<code>\tableofcontents</code> .....	21
<code>\ltx@ifempty</code> .....	229	<code>\Test</code> .....	310, 333
<code>\ltx@ifUndefined</code>	173, 187, 188, 204, 205	<code>\TestExpected</code> .....	342, 345, 352, 357
<code>\ltx@ifundefined</code> .....	175	<code>\TestResult</code> .....	343, 345, 354, 357
<b>M</b>		<code>\the</code> .....	132, 133, 134, 135, 136, 137, 138, 139, 152, 277, 297, 298
<code>\makeatletter</code> .....	4	<code>\TMP@EnsureCode</code> .....	149, 156, 157, 158, 159, 160, 161, 162, 163
<code>\makeatother</code> .....	19		
<b>N</b>		<b>U</b>	
<code>\newcommand</code> .	5, 16, 38, 39, 40, 191, 208	<code>\usepackage</code> .....	3, 37
<code>\next</code> .....	264, 266, 268		
<code>\number</code> .....	299	<b>W</b>	
<b>P</b>		<code>\write</code> .....	78, 107, 341, 346, 358
<code>\PackageInfo</code> .....	81	<b>X</b>	
<code>\protected</code> .....	195, 212	<code>\x</code> .....	69, 70, 73, 77, 81, 83, 106, 111, 121, 130, 142
<code>\ProvidesPackage</code> .....	74, 122		