

# The `engord` package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2008/08/11 v1.7

## Abstract

The package generates the suffix of English ordinal numbers. It can be used with plain and L<sup>A</sup>T<sub>E</sub>X formats.

## Contents

<b>1</b>	<b>Usage</b>	<b>1</b>
1.1	Package options	2
1.2	Examples	2
<b>2</b>	<b>Implementation</b>	<b>3</b>
2.1	Reload check and identification	3
2.2	Help commands for plain compatibility	4
2.3	User macros	4
2.4	Suffix generation	6
<b>3</b>	<b>Test</b>	<b>7</b>
3.1	Catcode checks for loading	7
<b>4</b>	<b>Installation</b>	<b>8</b>
4.1	Download	8
4.2	Bundle installation	9
4.3	Package installation	9
4.4	Refresh file name databases	9
4.5	Some details for the interested	9
<b>5</b>	<b>History</b>	<b>10</b>
	[2000/05/23 v1.0]	10
	[2003/04/28 v1.1]	10
	[2006/02/20 v1.2]	10
	[2007/04/11 v1.3]	10
	[2007/04/26 v1.4]	10
	[2007/09/09 v1.5]	10
	[2007/09/20 v1.6]	11
	[2008/08/11 v1.7]	11
<b>6</b>	<b>Index</b>	<b>11</b>

## 1 Usage

`\engord{<LATEX counter name>}`

It prints the value of the L<sup>A</sup>T<sub>E</sub>X counter as English ordinal number. It can be used in the same way as `\arabic`, `\roman`, or `\alph`. The command is not available in plain-T<sub>E</sub>X.

`\engordnumber {⟨any TEX number⟩}`

It prints the number as English ordinal number.

`\engordletters {#1}`

This command formats the English ordinal letters after the number. It defaults to `\textsuperscript`.

`\engorderror {#1}`

It can be redefined, if an other error handling is wanted. The argument is a negative number or zero.

`\engordraisetrue`  
`\engordraisefalse`

These commands set the switch `\ifengordraise` that is asked by the default `\engordletters` before raising the ordinal letters.

## 1.1 Package options

**normal:** `\engordraisefalse`

**raise:** `\engordraisetrue`

Default is `raise`.

## 1.2 Examples

- `\usepackage[normal]{engord}`  
`\engordnumber{1}` → 1st  
`\engordnumber{12}` → 12th  
`\engordnumber{123}` → 123rd  
`\engord{page}` → 1st (if page has the value of one)  
`\engordraisetrue`  
`\engordnumber{12}` → 12<sup>th</sup>

- The default output of a counter can be redefined:

```
\newcounter{mycounter}
\renewcommand{\theengcounter}{\engord{mycounter}}
```

- Because the implementation of `\engord` and `\engordnumber` is kept expandable, these commands can be used to make command names with an appropriate definition of `\engordletters`:

```
\renewcommand*{\engordletters}[1]{#1}
\@namedef{My\engordnumber{3}Command}{...}
```

This generates the command name ‘`\My4rdCommand`’. Since version 1.2 the redefinition can be dropped if the letters are not raised.

- If the letters should not be raised, use L<sup>A</sup>T<sub>E</sub>X package option `normal` or use

```
\engordraisefalse
```

Also `\engordletters` could be redefined for this purpose:

```
\renewcommand*{\engordletters}[1]{#1}
```

## 2 Implementation

### 2.1 Reload check and identification

```
1 (*package)
```

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

```
2 \begingroup
3  \catcode44 12 % ,
4  \catcode45 12 % -
5  \catcode46 12 % .
6  \catcode58 12 % :
7  \catcode64 11 % @
8  \catcode123 1 % {
9  \catcode125 2 % }
10 \expandafter\let\expandafter\x\csname ver@engord.sty\endcsname
11 \ifx\x\relax % plain-TEX, first loading
12 \else
13   \def\empty{}%
14   \ifx\x\empty % LATEX, first loading,
15     % variable is initialized, but \ProvidesPackage not yet seen
16   \else
17     \catcode35 6 % #
18     \expandafter\ifx\csname PackageInfo\endcsname\relax
19       \def\x#1#2{%
20         \immediate\write-1{Package #1 Info: #2.}%
21       }%
22     \else
23       \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24     \fi
25     \x{engord}{The package is already loaded}%
26   \aftergroup\endinput
27 \fi
28 \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31  \catcode35 6 % #
32  \catcode40 12 % (
33  \catcode41 12 % )
34  \catcode44 12 % ,
35  \catcode45 12 % -
36  \catcode46 12 % .
37  \catcode47 12 % /
38  \catcode58 12 % :
39  \catcode64 11 % @
40  \catcode91 12 % [
41  \catcode93 12 % ]
42  \catcode123 1 % {
43  \catcode125 2 % }
44  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45    \def\x#1#2#3[#4]{\endgroup
46      \immediate\write-1{Package: #3 #4}%
47      \xdef#1{#4}%
48    }%
49  \else
50    \def\x#1#2[#3]{\endgroup
51      #2[#{#3}]%
52      \ifx#1\@undefined
53        \xdef#1{#3}%
54      \fi
55      \ifx#1\relax
56        \xdef#1{#3}%
57      \fi
58    \endgroup
```

```

57      \fi
58    }%
59  \fi
60 \expandafter\x\csname ver@engord.sty\endcsname
61 \ProvidesPackage{engord}%
62 [2008/08/11 v1.7 Provides English ordinal numbers (H0)]

```

## 2.2 Help commands for plain compatibility

```

63 \begingroup
64   \catcode123 1 % {
65   \catcode125 2 % }
66   \def\x{\endgroup
67     \expandafter\edef\csname EO@AtEnd\endcsname{%
68       \catcode35 \the\catcode35\relax
69       \catcode64 \the\catcode64\relax
70       \catcode123 \the\catcode123\relax
71       \catcode125 \the\catcode125\relax
72     }%
73   }%
74 \x
75 \catcode35 6 % #
76 \catcode64 11 % @
77 \catcode123 1 % {
78 \catcode125 2 % }
79 \def\TMP@EnsureCode#1#2{%
80   \edef\EO@AtEnd{%
81     \EO@AtEnd
82     \catcode#1 \the\catcode#1\relax
83   }%
84   \catcode#1 #2\relax
85 }
86 \TMP@EnsureCode{33}{12}% !
87 \TMP@EnsureCode{36}{3}% $
88 \TMP@EnsureCode{39}{12}% '
89 \TMP@EnsureCode{42}{12}% *
90 \TMP@EnsureCode{46}{12}% .
91 \TMP@EnsureCode{47}{12}% /
92 \TMP@EnsureCode{60}{12}% <
93 \TMP@EnsureCode{94}{7}% ^ (superscript)
94 \TMP@EnsureCode{96}{12}% '

```

\EO@def Definitions, \newcommand does not exist in plain-TeX.

```

95 \begingroup\expandafter\expandafter\expandafter\endgroup
96 \expandafter\ifx\csname newcommand\endcsname\relax
97   \def\EO@def{\def}%
98 \else
99   \def\EO@def#1{%
100     \newcommand*{#1}{}%
101     \def#1%
102   }%
103 \fi

104 \begingroup\expandafter\expandafter\expandafter\endgroup
105 \expandafter\ifx\csname RequirePackage\endcsname\relax
106   \input infwarerr.sty\relax
107 \else
108   \RequirePackage{infwarerr}[2007/09/09]%
109 \fi

```

## 2.3 User macros

\ifengordraise The switch \ifengordraise, whether the ordinal letters are raised or not. Default is raised because of compatibility.

```

110 \newif\ifengordraise
111 \engordraisetrue

```

In L<sup>A</sup>T<sub>E</sub>X this also can be controlled by option normal or raise.

```

112 \begingroup\expandafter\expandafter\expandafter\endgroup
113 \expandafter\ifx\csname DeclareOption\endcsname\relax
114 \else
115   \DeclareOption{normal}{\engordraisefalse}%
116   \DeclareOption{raise}{\engordraisetrue}%
117   \ProcessOptions*\relax
118 \fi

```

**\engordletters** **\engordletters** is called with one argument, the english ordinal letters, and contains the code to format them. It defaults to **\textsuperscript** depending on **\ifengordraise**.

```

119 \expandafter\ifx\csname engordletters\endcsname\relax
120   \EO@def\engordletters{%
121     \ifengordraise
122       \expandafter\engordtextsuperscript
123     \fi
124   }%
125 \fi

```

**\engordtextsuperscript** For plain-T<sub>E</sub>X the definition is quite ugly, redefine **\engordtextsuperscript** if you have a better one.

```

126 \expandafter\ifx\csname engordtextsuperscript\endcsname\relax
127   \begingroup\expandafter\expandafter\expandafter\endgroup
128   \expandafter\ifx\csname textsuperscript\endcsname\relax
129     \def\engordtextsuperscript#1{%
130       \relax
131       \ifmmode
132         ^{\rm#1}%
133       \else
134         ${\rm#1}$%
135       \fi
136     }%
137   \else
138     \def\engordtextsuperscript{\textsuperscript}%
139   \fi
140 \fi

```

**\engorderror** **\engorderror** is called, if the number is zero or negative.

```

141 \expandafter\ifx\csname engorderror\endcsname\relax
142   \EO@def\engorderror#1{%
143     #1\engordletters{!ERROR!}%
144     \@PackageWarning{engord}{%
145       ‘#1’ is not an ordinal number%
146     }%
147   }%
148 \fi

```

**\engord** **\engord** expects a L<sup>A</sup>T<sub>E</sub>X counter name as argument and calls **\engordnumber**. It is defined only, if L<sup>A</sup>T<sub>E</sub>X is used.

```

149 \begingroup\expandafter\expandafter\expandafter\endgroup
150 \expandafter\ifx\csname newcounter\endcsname\relax
151 \else
152   \EO@def\engord#1{%
153     \engordnumber{\value{#1}}%
154   }%
155 \fi

```

`\engordnumber` `\engordnumber` is the user command to print a number as english ordinal number. The argument can be any T<sub>E</sub>X number like explicit numbers, register values, ...

In a safe way it converts the T<sub>E</sub>X number argument into a form that only consists of decimal digits.

```
156 \EO@def\engordnumber#1{%
157   \expandafter\EO@number\expandafter{\number#1}%
158 }
```

## 2.4 Suffix generation

`\EO@number` `\EO@number` expects a number with decimal digits as argument and looks at the size of the number and the count of the digits:

```
159 \def\EO@number#1{%
160   \ifnum#1<1 % handle the error case
161     \engorderror{#1}%
162   \else
163     \ifnum#1<21 %
164       \EO@ord{#1}%
165     \else
166       \ifnum#1<100 %
167         \EO@twodigits#1%
168       \else
169         \@ReturnAfterFi{%
170           \EO@reverse#1\@nil}\EO@afterreverse
171         }%
172       \fi
173     \fi
174   \fi
175 }
```

`\@ReturnAfterFi` An internal help macro to prevent a too deep `\if` nesting.

```
176 \long\def\@ReturnAfterFi#1\fi{\fi#1}
```

`\EO@ord` `\EO@ord` prints the number with ord letters.

`#1`: decimal digits, `#1 < 21`

```
177 \def\EO@ord#1{%
178   #1%
179   \expandafter\engordletters
180   \ifcase#1{th}\or
181     {st}\or
182     {nd}\or
183     {rd}\else
184     {th}%
185   \fi
186 }
```

`\EO@twodigits` `\EO@twodigits` expects a number with two digits, `20 < number < 100`

```
187 \def\EO@twodigits#1#2{%
188   #1\EO@ord{#2}%
189 }
```

`\EO@reverse` `\EO@reverse` reverses the digits of the number.

`#1`: next digit

`#2`: rest of the digits

`#3`: already reversed digits

`#4`: next command to call with the reversed number as argument

```
190 \def\EO@reverse#1#2\@nil#3#4{%
191   \ifx\\#2\\%
192     #4{#1#3}%

```

```

193 \else
194   \@ReturnAfterFi{%
195     \EO@reverse#2\@nil{#1#3}{#4}%
196   }%
197 \fi
198 }

\EO@afterreverse \EO@afterreverse calls \EO@reverseback so that \EO@reverseback can inspect
the digits of the number.

199 \def\EO@afterreverse#1{%
200   \EO@reverseback#1\@nil
201 }

\EO@reverseback \EO@reverseback reverses the reversion.
#1: the last digit of the number
#2: the second last digit of the number
#3: first digits of the number in reversed order, it is not empty, because
\EO@reverseback is only called with numbers > 100.

202 \def\EO@reverseback#1#2#3\@nil{%
203   \EO@reverse#3\@nil{}\@firstofone
204   \ifnum#2#1<21 %
205     \EO@ord{#2#1}%
206   \else
207     #2\EO@ord{#1}%
208   \fi
209 }

210 \EO@AtEnd
211 </package>

```

## 3 Test

### 3.1 Catcode checks for loading

```

212 <*test1>
213 \catcode'\{=1 %
214 \catcode'\}=2 %
215 \catcode'\#=6 %
216 \catcode'\@=11 %
217 \expandafter\ifx\csname count@\endcsname\relax
218   \countdef\count@=255 %
219 \fi
220 \expandafter\ifx\csname @gobble\endcsname\relax
221   \long\def\@gobble#1{%
222 \fi
223 \expandafter\ifx\csname @firstofone\endcsname\relax
224   \long\def\@firstofone#1{#1}%
225 \fi
226 \expandafter\ifx\csname loop\endcsname\relax
227   \expandafter\@firstofone
228 \else
229   \expandafter\@gobble
230 \fi
231 {%
232   \def\loop#1\repeat{%
233     \def\body{#1}%
234     \iterate
235   }%
236   \def\iterate{%
237     \body
238     \let\next\iterate

```

```

239     \else
240     \let\next\relax
241     \fi
242     \next
243 }%
244 \let\repeat=\fi
245 }%
246 \def\RestoreCatcodes{}
247 \count@=0 %
248 \loop
249   \edef\RestoreCatcodes{%
250     \RestoreCatcodes
251     \catcode\the\count@=\the\catcode\count@\relax
252   }%
253 \ifnum\count@<255 %
254   \advance\count@ 1 %
255 \repeat
256
257 \def\RangeCatcodeInvalid#1#2{%
258   \count@=#1\relax
259   \loop
260     \catcode\count@=15 %
261     \ifnum\count@<#2\relax
262       \advance\count@ 1 %
263     \repeat
264 }
265 \expandafter\ifx\csname LoadCommand\endcsname\relax
266   \def\LoadCommand{\input engord.sty\relax}%
267 \fi
268 \def\Test{%
269   \RangeCatcodeInvalid{0}{47}%
270   \RangeCatcodeInvalid{58}{64}%
271   \RangeCatcodeInvalid{91}{96}%
272   \RangeCatcodeInvalid{123}{255}%
273   \catcode'\@=12 %
274   \catcode'\=0 %
275   \catcode'\{=1 %
276   \catcode'\}=2 %
277   \catcode'\#=6 %
278   \catcode'\[=12 %
279   \catcode'\]=12 %
280   \catcode'\%=14 %
281   \catcode'\ =10 %
282   \catcode13=5 %
283   \LoadCommand
284   \RestoreCatcodes
285 }
286 \Test
287 \csname @@end\endcsname
288 \end
289 </test1>

```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

[CTAN:macros/latex/contrib/oberdiek/engord.dtx](http://ctan.org/macros/latex/contrib/oberdiek/engord.dtx) The source file.

[CTAN:macros/latex/contrib/oberdiek/engord.pdf](http://ctan.org/macros/latex/contrib/oberdiek/engord.pdf) Documentation.

---

<sup>1</sup>[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)



**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/
```

## 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain-T<sub>E</sub>X:

```
tex engord.dtx
```

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

<code>engord.sty</code>	→ <code>tex/generic/oberdiek/engord.sty</code>
<code>engord.pdf</code>	→ <code>doc/latex/oberdiek/engord.pdf</code>
<code>test/engord-test1.tex</code>	→ <code>doc/latex/oberdiek/test/engord-test1.tex</code>
<code>engord.dtx</code>	→ <code>source/latex/oberdiek/engord.dtx</code>

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 T<sub>E</sub>X distribution (teT<sub>E</sub>X, miK<sub>T</sub>E<sub>X</sub>, ...) relies on file name databases, you must refresh these. For example, teT<sub>E</sub>X users run `texhash` or `mktexlsr`.

## 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 engord.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{engord.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 engord.dtx
makeindex -s gind.ist engord.idx
pdflatex engord.dtx
makeindex -s gind.ist engord.idx
pdflatex engord.dtx
```

## 5 History

[2000/05/23 v1.0]

- First public release, published in newsgroup [de.comp.text.tex](#):  
“[Re: Ordinalzahlen in LaTeX?](#)”<sup>2</sup>

[2003/04/28 v1.1]

- Bug fix for 30, 40, 50, ..., 100, 130, ...
- \ordletters renamed to documented \engordletters.

[2006/02/20 v1.2]

- Support for plain-T<sub>E</sub>X.
- Switch \ifengordraise added.
- Package options raise and normal added.
- DTX framework.

[2007/04/11 v1.3]

- Line ends sanitized.

[2007/04/26 v1.4]

- Use of package infwarerr.

[2007/09/09 v1.5]

- Catcode section added.

---

<sup>2</sup>Url: <http://groups.google.com/group/de.comp.text.tex/msg/738e2cb4c51759d6>

[2007/09/20 v1.6]

- Short description fixed (George White).

[2008/08/11 v1.7]

- Code is not changed.
- URLs updated.

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

Symbols	
<code>\#</code> .....	215, 277
<code>\%</code> .....	280
<code>\@</code> .....	216, 273
<code>\@PackageWarning</code> .....	144
<code>\@ReturnAfterFi</code> .....	169, <u>176</u> , 194
<code>\@firstofone</code> .....	203, 224, 227
<code>\@gobble</code> .....	221, 229
<code>\@nil</code> .....	170, 190, 195, 200, 202, 203
<code>\@undefined</code> .....	52
<code>\[</code> .....	278
<code>\]</code> .....	191, 274
<code>\{</code> .....	213, 275
<code>\}</code> .....	214, 276
<code>\]</code> .....	279
<code>\_</code> .....	281
A	
<code>\advance</code> .....	254, 262
<code>\aftergroup</code> .....	26
B	
<code>\body</code> .....	233, 237
C	
<code>\catcode</code> 3, 4, 5, 6, 7, 8, 9, 17, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 64, 65, 68, 69, 70, 71, 75, 76, 77, 78, 82, 84, 213, 214, 215, 216, 251, 260, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282	
<code>\count@</code> .....	218, 247, 251, 253, 254, 258, 260, 261, 262
<code>\countdef</code> .....	218
<code>\csname</code> .....	10, 18, 44, 60, 67, 96, 105, 113, 119, 126, 128, 141, 150, 217, 220, 223, 226, 265, 287
D	
<code>\DeclareOption</code> .....	115, 116
E	
<code>\empty</code> .....	13, 14
<code>\end</code> .....	288
<code>\endcsname</code> .....	10, 18, 44, 60, 67, 96, 105, 113, 119, 126, 128, 141, 150, 217, 220, 223, 226, 265, 287
<code>\endinput</code> .....	26
<code>\engord</code> .....	<u>1</u> , <u>149</u>
<code>\engorderror</code> .....	2, <u>141</u> , 161
<code>\engordletters</code> .....	2, <u>119</u> , 143, 179
<code>\engordnumber</code> .....	2, 153, <u>156</u>
<code>\engordraisefalse</code> .....	2, 115
<code>\engordraisetrue</code> .....	2, 111, 116
<code>\engordtextsuperscript</code> .....	122, <u>126</u>
<code>\EO@afterreverse</code> .....	170, <u>199</u>
<code>\EO@AtEnd</code> .....	80, 81, 210
<code>\EO@def</code> .....	<u>95</u> , 120, 142, 152, 156
<code>\EO@number</code> .....	157, <u>159</u>
<code>\EO@ord</code> .....	164, <u>177</u> , 188, 205, 207
<code>\EO@reverse</code> .....	170, <u>190</u> , 203
<code>\EO@reverseback</code> .....	200, <u>202</u>
<code>\EO@twodigits</code> .....	167, <u>187</u>
I	
<code>\ifcase</code> .....	180
<code>\ifengordraise</code> .....	<u>110</u> , 121
<code>\ifmmode</code> .....	131
<code>\ifnum</code> .....	160, 163, 166, 204, 253, 261
<code>\ifx</code> .....	11, 14, 18, 44, 52, 55, 96, 105, 113, 119, 126, 128, 141, 150, 191, 217, 220, 223, 226, 265
<code>\immediate</code> .....	20, 46
<code>\input</code> .....	106, 266
<code>\iterate</code> .....	234, 236, 238
L	
<code>\LoadCommand</code> .....	266, 283
<code>\loop</code> .....	232, 248, 259
N	
<code>\newcommand</code> .....	100
<code>\newif</code> .....	110
<code>\next</code> .....	238, 240, 242
<code>\number</code> .....	157
P	
<code>\PackageInfo</code> .....	23
<code>\ProcessOptions</code> .....	117
<code>\ProvidesPackage</code> .....	15, 61

<b>R</b>		<code>\TMP@EnsureCode</code> . . . . . 79,
<code>\RangeCatcodeInvalid</code> . . . . .	257, 269, 270, 271, 272	86, 87, 88, 89, 90, 91, 92, 93, 94
<code>\repeat</code> . . . . .	232, 244, 255, 263	<b>V</b>
<code>\RequirePackage</code> . . . . .	108	<code>\value</code> . . . . . 153
<code>\RestoreCatcodes</code> . . . . .	246, 249, 250, 284	<b>W</b>
<code>\rm</code> . . . . .	132, 134	<code>\write</code> . . . . . 20, 46
<b>T</b>		<b>X</b>
<code>\Test</code> . . . . .	268, 286	<code>\x</code> 10, 11, 14, 19, 23, 25, 45, 50, 60, 66, 74
<code>\textsuperscript</code> . . . . .	138	
<code>\the</code> . . . . .	68, 69, 70, 71, 82, 251	