The engord package

Heiko Oberdiek
<heiko.oberdiek at googlemail.com>

2010/03/01 v1.8

Abstract

The package generates the suffix of English ordinal numbers. It can be used with plain and LaTeX formats.

Contents

1 Usage **2**
1.1 Package options . . . 2
1.2 Examples . . . 2

2 Implementation **3**
2.1 Reload check and identification . . . 3
2.2 Help commands for plain compatibility . . . 4
2.3 User macros . . . 5
2.4 Suffix generation . . . 6

3 Test **7**
3.1 Catcode checks for loading . . . 7

4 Installation **9**
4.1 Download . . . 9
4.2 Bundle installation . . . 9
4.3 Package installation . . . 9
4.4 Refresh file name databases . . . 9
4.5 Some details for the interested . . . 10

5 History **10**
[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] . . . 11
[2007/09/09 v1.5] . . . 11
[2007/09/20 v1.6] . . . 11
[2008/08/11 v1.7] . . . 11
[2010/03/01 v1.8] . . . 11

6 Index **11**

1 Usage

`\engord{`⟨*LaTeX counter name*⟩`}`

It prints the value of the LaTeX 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 TeX.

`\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}  → 12th
  ```

- 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 LaTeX 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 LaTeX.

```
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
```

```
      \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@engord.sty\endcsname
\ProvidesPackage{engord}%
  [2010/03/01 v1.8 Provides English ordinal numbers (HO)]
```

2.2 Help commands for plain compatibility

```
\begingroup
  \catcode123 1 % {
  \catcode125 2 % }
  \def\x{\endgroup
    \expandafter\edef\csname EO@AtEnd\endcsname{%
      \catcode35 \the\catcode35\relax
      \catcode64 \the\catcode64\relax
      \catcode123 \the\catcode123\relax
      \catcode125 \the\catcode125\relax
    }%
  }%
\x
\catcode35 6 % #
\catcode64 11 % @
\catcode123 1 % {
\catcode125 2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\EO@AtEnd{%
    \EO@AtEnd
    \catcode#1 \the\catcode#1\relax
  }%
  \catcode#1 #2\relax
}
\TMP@EnsureCode{33}{12}% !
\TMP@EnsureCode{36}{3}% $
\TMP@EnsureCode{39}{12}% '
\TMP@EnsureCode{42}{12}% *
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{60}{12}% <
\TMP@EnsureCode{94}{7}% ^(superscript)
\TMP@EnsureCode{96}{12}% `
```

`\EO@def` Definitions, `\newcommand` does not exist in plain TEX.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname newcommand\endcsname\relax
  \def\EO@def{\def}%
\else
  \def\EO@def#1{%
    \newcommand*{#1}{}%
    \def#1%
  }%
\fi
```

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input infwarerr.sty\relax
  \input ltxcmds.sty\relax
\else
  \RequirePackage{infwarerr}[2007/09/09]%
  \RequirePackage{ltxcmds}[2010/03/01]%
\fi
```

2.3 User macros

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

```
\ltx@newif\ifengordraise
\engordraisetrue
```

In LaTeX this also can be controlled by option normal or raise.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname DeclareOption\endcsname\relax
\else
  \DeclareOption{normal}{\engordraisefalse}%
  \DeclareOption{raise}{\engordraisetrue}%
  \ProcessOptions*\relax
\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`.

```
\expandafter\ifx\csname engordletters\endcsname\relax
  \EO@def\engordletters{%
    \ifengordraise
      \expandafter\engordtextsuperscript
    \fi
  }%
\fi
```

`\engordtextsuperscript` For plain TeX the definition is quite ugly, redefine `\engordtextsuperscript` if you have a better one.

```
\expandafter\ifx\csname engordtextsuperscript\endcsname\relax
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname textsuperscript\endcsname\relax
    \def\engordtextsuperscript#1{%
      \relax
      \ifmmode
        ^{\rm#1}%
      \else
        $^{\rm#1}$%
      \fi
    }%
  \else
    \def\engordtextsuperscript{\textsuperscript}%
  \fi
\fi
```

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

```
\expandafter\ifx\csname engorderror\endcsname\relax
  \EO@def\engorderror#1{%
    #1\engordletters{!ERROR!}%
    \@PackageWarning{engord}{%
      `#1' is not an ordinal number%
    }%
```

```
  }%
\fi
```

`\engord` `\engord` expects a LaTeX counter name as argument and calls `\engordnumber`. It is defined only, if LaTeX is used.

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname newcounter\endcsname\relax
\else
  \EO@def\engord#1{%
    \engordnumber{\value{#1}}%
  }%
\fi
```

`\engordnumber` `\engordnumber` is the user command to print a number as english ordinal number. The argument can be any TeX number like explicit numbers, register values, ...

In a safe way it converts the TeX number argument into a form that only consists of decimal digits.

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

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:

```
\def\EO@number#1{%
  \ifnum#1<1 % handle the error case
    \engorderror{#1}%
  \else
    \ifnum#1<21 %
      \EO@ord{#1}%
    \else
      \ifnum#1<100 %
        \EO@twodigits#1%
      \else
        \@ReturnAfterFi{%
          \EO@reverse#1\@nil{}\EO@afterreverse
        }%
      \fi
    \fi
  \fi
}
```

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

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

`\EO@ord` `\EO@ord` prints the number with ord letters.
`#1`: decimal digits, `#1` < 21

```
\def\EO@ord#1{%
  #1%
  \expandafter\engordletters
  \ifcase#1{th}\or
    {st}\or
    {nd}\or
    {rd}\else
    {th}%
  \fi
}
```

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

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

`\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

```
\def\EO@reverse#1#2\@nil#3#4{%
  \ifx\\#2\\%
    #4{#1#3}%
  \else
    \@ReturnAfterFi{%
      \EO@reverse#2\@nil{#1#3}{#4}%
    }%
  \fi
}
```

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

```
\def\EO@afterreverse#1{%
  \EO@reverseback#1\@nil
}
```

`\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.

```
\def\EO@reverseback#1#2#3\@nil{%
  \EO@reverse#3\@nil{}\@firstofone
  \ifnum#2#1<21 %
    \EO@ord{#2#1}%
  \else
    #2\EO@ord{#1}%
  \fi
}

\EO@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
}
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input engord.sty\relax}%
\fi
\def\Test{%
  \RangeCatcodeInvalid{0}{47}%
  \RangeCatcodeInvalid{58}{64}%
  \RangeCatcodeInvalid{91}{96}%
  \RangeCatcodeInvalid{123}{255}%
  \catcode`\@=12 %
  \catcode`\\=0 %
  \catcode`\{=1 %
  \catcode`\}=2 %
  \catcode`\#=6 %
  \catcode`\[=12 %
  \catcode`\]=12 %
  \catcode`\%=14 %
  \catcode`\ =10 %
  \catcode13=5 %
  \LoadCommand
  \RestoreCatcodes
}
\Test
\csname @@end\endcsname
```

```
290 \end
291 ⟨/test1⟩
```

4 Installation

4.1 Download

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

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

CTAN:macros/latex/contrib/oberdiek/engord.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 engord.dtx
```

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

```
engord.sty            → tex/generic/oberdiek/engord.sty
engord.pdf            → doc/latex/oberdiek/engord.pdf
test/engord-test1.tex → doc/latex/oberdiek/test/engord-test1.tex
engord.dtx            → source/latex/oberdiek/engord.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 engord.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{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 pdfLaTeX:

```
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?"[2]

[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 TeX.
- Switch `\ifengordraise` added.
- Package options raise and normal added.
- DTX framework.

[2007/04/11 v1.3]

- Line ends sanitized.

[2] Url: http://groups.google.com/group/de.comp.text.tex/msg/738e2cb4c51759d6

[2007/04/26 v1.4]

- Use of package infwarerr.

[2007/09/09 v1.5]

- Catcode section added.

[2007/09/20 v1.6]

- Short description fixed (George White).

[2008/08/11 v1.7]

- Code is not changed.
- URLs updated.

[2010/03/01 v1.8]

- Compatibility with ini-TEX.

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.

Symbols

\# 217, 279
\% 282
\@ 218, 275
\@PackageWarning 146
\@ReturnAfterFi 171, 178, 196
\@firstofone 205, 226, 229
\@gobble 223, 231
\@nil 172, 192, 197, 202, 204, 205
\@undefined 52
\[. 280
\\ 193, 276
\{ 215, 277
\} 216, 278
\] 281

\␣ 283

A

\advance 256, 264
\aftergroup 26

B

\body 235, 239

C

\catcode 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, 215, 216, 217, 218, 253, 262, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284
\count@ 220, 249, 253, 255, 256, 260, 262, 263, 264
\countdef 220
\csname 10, 18, 44, 60, 67, 96, 105, 115, 121, 128, 130, 143, 152, 219, 222, 225, 228, 267, 289

D

\DeclareOption 117, 118

E

\empty 13, 14
\end 290
\endcsname 10, 18, 44, 60, 67, 96, 105, 115, 121, 128, 130, 143, 152, 219, 222, 225, 228, 267, 289
\endinput 26
\engord *2*, 151
\engorderror *2*, 143, 163
\engordletters *2*, 121, 145, 181
\engordnumber *2*, 155, 158
\engordraisefalse *2*, 117
\engordraisetrue *2*, 113, 118
\engordtextsuperscript 124, 128
\EO@afterreverse 172, 201
\EO@AtEnd 80, 81, 212
\EO@def 95, 122, 144, 154, 158
\EO@number 159, 161
\EO@ord 166, 179, 190, 207, 209
\EO@reverse 172, 192, 205
\EO@reverseback 202, 204
\EO@twodigits 169, 189

I

\ifcase 182
\ifengordraise 112, 123
\ifmmode 133
\ifnum 162, 165, 168, 206, 255, 263
\ifx 11, 14, 18, 44, 52, 55, 96, 105, 115, 121, 128, 130, 143, 152, 193, 219, 222, 225, 228, 267
\immediate 20, 46
\input 106, 107, 268
\iterate 236, 238, 240

L

\LoadCommand 268, 285
\loop 234, 250, 261
\ltx@newif 112

N

\newcommand 100
\next 240, 242, 244
\number 159

P

\PackageInfo 23
\ProcessOptions 119
\ProvidesPackage 15, 61

R

\RangeCatcodeInvalid 259, 271, 272, 273, 274
\repeat 234, 246, 257, 265
\RequirePackage 109, 110
\RestoreCatcodes . . 248, 251, 252, 286
\rm 134, 136

T

\Test 270, 288
\textsuperscript 140
\the 68, 69, 70, 71, 82, 253
\TMP@EnsureCode 79, 86, 87, 88, 89, 90, 91, 92, 93, 94

V

\value 155

W

\write 20, 46

X

\x 10, 11, 14, 19, 23, 25, 45, 50, 60, 66, 74