The eolgrab package

Heiko Oberdiek
<heiko.oberdiek at googlemail.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

1 Documentation

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

[1] Info page for mailing list texhax: http://tug.org/mailman/listinfo/texhax

```
\eolsection My Title
```

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 TEX and 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 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 ε-TEX's `\protected` or 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 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 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 LaTeX:

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

Example for plain TeX:

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

And a sophisticated variant for LaTeX that also supports the star syntax and the optional argument:

```
⟨*example-sec⟩
\documentclass{article}
\usepackage{eolgrab}
\makeatletter
\newcommand*{\eolsection}{%
  \@ifstar{%
    \eolgrab{\section*}%
  }{%
    \@ifnextchar[{%
      \eoloptsection
    }{%
      \eolgrab\section
    }%
  }%
}
\newcommand*{\eoloptsection}[1][]{%
  \eolgrab{\section[{#1}]}%
}
\makeatother
\begin{document}
\tableofcontents
\eolsection Section without star and optional argument
\eolsection*Section with star
\eolsection[Short section title]Long section title
\end{document}
⟨/example-sec⟩
```

1.1.1 Small LaTeX document as example

```
⟨*example-ltx⟩
\RequirePackage{eolgrab}
\eolgrab\documentclass article
\eolgrab\begin document
\eolgrab\section Hello World
\eolgrab\emph Some text
\eolgrab\end document
⟨/example-ltx⟩
```

1.1.2 LaTeX 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 TeX 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 TeX and LaTeX the standard settings of the inserted endline character is the character with code 13 (or `^^M` in TeX 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 LaTeX.

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

```
\endlinechar=13 %
\catcode35=6 % #
\catcode39=12 % '
\catcode44=12 % ,
\catcode45=12 % -
\catcode46=12 % .
\catcode58=12 % :
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\expandafter\let\expandafter\x\csname ver@eolgrab.sty\endcsname
\ifx\x\relax % plain-TeX, first loading
\else
  \def\empty{}%
  \ifx\x\empty % LaTeX, first loading,
    % variable is initialized, but \ProvidesPackage not yet seen
  \else
    \expandafter\ifx\csname PackageInfo\endcsname\relax
      \def\x#1#2{%
        \immediate\write-1{Package #1 Info: #2.}%
      }%
    \else
      \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
    \fi
    \x{eolgrab}{The package is already loaded}%
    \aftergroup\endinput
  \fi
\fi
\endgroup%
```

Package identification:

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode35=6 % #
  \catcode39=12 % '
  \catcode40=12 % (
  \catcode41=12 % )
  \catcode44=12 % ,
  \catcode45=12 % -
  \catcode46=12 % .
  \catcode47=12 % /
  \catcode58=12 % :
  \catcode64=11 % @
  \catcode91=12 % [
  \catcode93=12 % ]
  \catcode123=1 % {
  \catcode125=2 % }
  \expandafter\ifx\csname ProvidesPackage\endcsname\relax
    \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@eolgrab.sty\endcsname
\ProvidesPackage{eolgrab}%
  [2011/01/12 v1.0 Catch arguments delimited by end of line (HO)]%
```

2.2 Catcodes

```
\begingroup\catcode61\catcode48\catcode32=10\relax%
  \catcode13=5 % ^^M
  \endlinechar=13 %
  \catcode123=1 % {
  \catcode125=2 % }
  \catcode64=11 % @
  \def\x{\endgroup
    \expandafter\edef\csname eolgrab@AtEnd\endcsname{%
      \endlinechar=\the\endlinechar\relax
      \catcode13=\the\catcode13\relax
      \catcode32=\the\catcode32\relax
      \catcode35=\the\catcode35\relax
      \catcode61=\the\catcode61\relax
      \catcode64=\the\catcode64\relax
      \catcode123=\the\catcode123\relax
      \catcode125=\the\catcode125\relax
    }%
  }%
\x\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\TMP@EnsureCode#1#2{%
  \edef\eolgrab@AtEnd{%
    \eolgrab@AtEnd
    \catcode#1=\the\catcode#1\relax
  }%
  \catcode#1=#2\relax
}
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{42}{12}% *
\TMP@EnsureCode{46}{12}% .
\TMP@EnsureCode{47}{12}% /
\TMP@EnsureCode{91}{12}% [
\TMP@EnsureCode{93}{12}% ]
\TMP@EnsureCode{94}{7}% ^
\edef\eolgrab@AtEnd{\eolgrab@AtEnd\noexpand\endinput}
```

2.3 Resources

```
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname RequirePackage\endcsname\relax
  \input ltxcmds.sty\relax
  \input infwarerr.sty\relax
\else
  \RequirePackage{ltxcmds}[2010/12/04]%
  \RequirePackage{infwarerr}[2010/04/08]%
\fi
```

`\eolgrab@ifdefinable`

```
\ltx@IfUndefined{@ifdefinable}{%
  \def\eolgrab@ifdefinable#1#2{%
```

```
    \ltx@ifundefined{#1}{#2}{%
      \@PackageError{eolgrab}{%
        Command \ltx@backslashchar#1 already defined%
      }\@ehc
    }%
  }%
}{%
  \def\eolgrab@ifdefinable#1{%
    \expandafter\@ifdefinable\csname#1\endcsname
  }%
}
```

2.4 Macro \eolgrab

\eolgrab

```
\eolgrab@ifdefinable{eolgrab}{%
  \ltx@IfUndefined{protected}{%
    \ltx@IfUndefined{DeclareRobustCommand}{%
      \def\eolgrab#1%
    }{%
      \newcommand\eolgrab{}%
      \DeclareRobustCommand*\eolgrab
    }%
  }{%
    \protected\def\eolgrab#1%
  }{%
    \begingroup
    \endlinechar=13 %
    \catcode13=\ltx@active
    \eolgrab@{#1}%
  }%
}
```

\eolgrabopt

```
\eolgrab@ifdefinable{eolgrabopt}{%
  \ltx@IfUndefined{protected}{%
    \ltx@IfUndefined{DeclareRobustCommand}{%
      \def\eolgrabopt#1%
    }{%
      \newcommand\eolgrabopt{}%
      \DeclareRobustCommand*\eolgrabopt
    }%
  }{%
    \protected\def\eolgrabopt#1%
  }{%
    \begingroup
    \endlinechar=13 %
    \catcode13=\ltx@active
    \eolgrab@opt{#1}%
  }%
}

\begingroup
  \catcode13=\ltx@active %
\ltx@firstofone{\endgroup %
```

\eolgrab@

```
  \def\eolgrab@#1#2^^M{%
    \endgroup %
    #1{#2}%
  }%
```

\eolgrab@opt

```
  \def\eolgrab@opt#1#2^^M{%
    \endgroup %
    \ltx@ifempty{#2}{%
      #1%
    }{%
      #1[{#2}]%
    }%
  }%
}%
\eolgrab@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
}
\def\RangeCatcodeCheck#1#2#3{%
  \count@=#1\relax
  \loop
    \ifnum#3=\catcode\count@
    \else
      \errmessage{%
        Character \the\count@\space
        with wrong catcode \the\catcode\count@\space
        instead of \number#3%
      }%
    \fi
  \ifnum\count@<#2\relax
    \advance\count@ 1 %
  \repeat
}
\def\space{ }
\expandafter\ifx\csname LoadCommand\endcsname\relax
  \def\LoadCommand{\input eolgrab.sty\relax}%
\fi
\def\Test{%
  \RangeCatcodeInvalid{0}{47}%
  \RangeCatcodeInvalid{58}{64}%
  \RangeCatcodeInvalid{91}{96}%
  \RangeCatcodeInvalid{123}{255}%
  \catcode`\@=12 %
  \catcode`\\=0 %
  \catcode`\%=14 %
  \LoadCommand
  \RangeCatcodeCheck{0}{36}{15}%
  \RangeCatcodeCheck{37}{37}{14}%
  \RangeCatcodeCheck{38}{47}{15}%
  \RangeCatcodeCheck{48}{57}{12}%
  \RangeCatcodeCheck{58}{63}{15}%
  \RangeCatcodeCheck{64}{64}{12}%
  \RangeCatcodeCheck{65}{90}{11}%
  \RangeCatcodeCheck{91}{91}{15}%
  \RangeCatcodeCheck{92}{92}{0}%
  \RangeCatcodeCheck{93}{96}{15}%
  \RangeCatcodeCheck{97}{122}{11}%
  \RangeCatcodeCheck{123}{255}{15}%
  \RestoreCatcodes
}
\Test
\csname @@end\endcsname
\end
```

⟨/test1⟩

3.2 Tests for plain TEX

```
⟨*test2⟩
\input eolgrab.sty\relax
\catcode`\{=1 %
\catcode`\}=2 %
```

```
\eolgrab{\immediate\write16}Hello World
\def\TestExpected{foo bar}
\eolgrab{\def\TestResult}foo bar

\ifx\TestExpected\TestResult
  \immediate\write16{* Ok (foo bar)}%
\else
  \errmessage{Test failed (foo bar)}%
\fi

\begingroup
  \def\TestExpected{foobar}%
  \endlinechar=-1 %
  \eolgrab{\def\TestResult}foo
  bar
  ^^M
  \ifx\TestExpected\TestResult
    \immediate\write16{* Ok (foobar)}%
  \else
    \errmessage{Test failed (foobar)}%
  \fi
\endgroup

\csname @@end\endcsname\end
⟨/test2⟩
```

4 Installation

4.1 Download

Package. This package is available on CTAN[2]:

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 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/
```

[2] ftp://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 (teTEX, mikTEX, ...) relies on file name databases, you must refresh these. For example, teTEX 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 eolgrab.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{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 pdfLATEX:

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

5 Catalogue

The following XML file can be used as source for the TEX 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 `eolgrab.xml`.

```
⟨*catalogue⟩
<?xml version='1.0' encoding='us-ascii'?>
<!DOCTYPE entry SYSTEM 'catalogue.dtd'>
<entry datestamp='$Date$' modifier='$Author$' id='eolgrab'>
  <name>eolgrab</name>
  <caption>Catch arguments delimited by end of line.</caption>
  <authorref id='auth:oberdiek'/>
  <copyright owner='Heiko Oberdiek' year='2011'/>
  <license type='lppl1.3'/>
  <version number='1.0'/>
  <description>
    The package defines the command <tt>\eolgrab</tt> that reads an
    string terminated by end of line, and then calls a macro with that
    string as argument.  Thus
    <pre>\eolgrab\section My title</pre> is equivalent to
    <pre>\section{My title}</pre>
    <p/>
    A second command <tt>\eolgrabopt</tt> provides the string as an
    optional argument (which is useful for <tt>\item</tt>, for example).
    <p/>
    The package is part of the <xref refid='oberdiek'>oberdiek</xref> bundle.
  </description>
  <documentation details='Package documentation'
      href='ctan:/macros/latex/contrib/oberdiek/eolgrab.pdf'/>
  <ctan file='true' path='/macros/latex/contrib/oberdiek/eolgrab.dtx'/>
  <miktex location='oberdiek'/>
  <texlive location='oberdiek'/>
  <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip'/>
</entry>
⟨/catalogue⟩
```

6 References

[1] Arno Trautmann, *[texhax] read argument until EOL*; mailing list `texthax@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.

7 History

[2011/01/12 v1.0]

- First public version.

8 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.