The compsci LaTeX package
macros for writing about macros (Frankenstein's conscience)

Matt Swift `<swift@alum.mit.edu>`

Version: 1.2   Date: 2001/08/31
Documentation revision: 2001/08/31

Abstract

Compsci is useful whenever writing about programming, but especially when writing about TEX and especially when used as a supplement to the *ltxdoc* class to document LATEX macros in a literate programming style with `dtx` files.

Contents

I Discussion **2**

1 Textual elements **2**

2 Referring to commands **4**

3 Miscellaneous **5**

II Implementation **6**

4 Version control **6**

5 Requirements and context **6**

6 Hyperlatex option **6**

7 Textual elements **8**

8 Referring to commands **13**

9 Miscellaneous **15**

III Configuration **17**

Part I

Discussion

1 Textual elements

`\env`

`\env {`⟨*environment name*⟩`}` typesets the name of a LaTeX environment. For example, the `enumerate` environment.

To do: *Add* `\bib` *how about?*

`\bst`

`\bst {`⟨*bibstyle name*⟩`}` typesets the name of a BibTeX bibliography style. For example, the `achicago` bibstyle.

`\package`

`\package {`⟨*package name*⟩`}` typesets the name of a LaTeX package. For example, the *compsci* package.

`\class`

`\class {`⟨*class name*⟩`}` typesets the name of a LaTeX class. For example, the *letter* class.

`\file`

`\file {`⟨*file name*⟩`}` typesets the name of a disk file. For example, the file `Makefile.in`.

To do: *Should I use url.sty's* `\path` *command for this?*

`\ext`

`\ext {`⟨*extension*⟩`}` typesets the name of a disk file extension. For example, the extension `dvi`. The argument should not include the dot that separates the root name from the extension. The dot can be inserted by the macro if you want it.

`\caveat`
`warning`

`\caveat {`⟨*warning text*⟩`}` typesets warning paragraphs. You can also enclose the ⟨*warning text*⟩ in a `warning` environment. Using the `warning` environment is necessary if the ⟨*warning text*⟩ is to include `|...|`s, a `codeexample` environment or other "verbatim" constructs.

Warning: *This is an example* `\caveat` *or* `warning` *environment.*

`\todo`
`todoenv`

The `\todo {`⟨*text*⟩`}` command and the `todoenv` environment are analogous to `\caveat` and `warning`, but they typeset paragraphs of a task "to do".

To do: *This is an example* `\todo` *or* `todoenv` *environment.*

`\code`
`\code*`
`|...|`

`\code {`⟨*verbatim text*⟩`}` and `|`⟨*verbatim text*⟩`|` typeset short pieces of code verbatim. For example, `|{{@#%$^\|` looks like `{{@#%$^\`.

Instead of matching curly braces, any two identical characters, even a space, may be used to contain the ⟨*verbatim*⟩ after the `\code` command.

To do: *Implement* `\code*`*. Like* `\verb*`*,* `\code*` *marks spaces explicitly. This is not an example todo, this is a real one for the compsci package!*

`\typeset`

`\typeset {`⟨*text*⟩`}` typesets ⟨*text*⟩ in a normal way, but indicates by boxing that the contents are an example of how some piece of code actually appears. For example, `this` looks like this when typeset.

`typesetexample`

These example environments may be used to present a piece of source code and what it looks like when typeset. The `typesetexample` environment typesets its contents like `\typeset` except they are set off in their own indented block element instead of being boxed.

> This is an example of the `typesetexample` environment.

`codeexample`
`codeexample*`
`codeexamplex`
`codeexamplex*`

The `codeexample` and `codeexample*` environments are similar analogs of `\code` and `\code*`. The `codeexamplex` environment is like the `codeexample` environment except that `\`, `{`, `}` have their usual meanings. Inside the argument, `\\` produces a typeset backslash.

```
This is a \env{codeexample} environment: \(\int e^x\,dx\)  {}{{{!#@$%^&\
There is a percent sign in the source after the $.
```

```
␣␣␣This␣is␣a␣\env{codexample*}␣environment:␣\(\int␣e^x\,dx\)␣␣{}{{{!#@$%^&\
␣␣␣Someday␣I␣would␣like␣to␣get␣rid␣of␣the␣leading␣spaces␣in␣a␣line.
␣␣␣
```

```
This is a codeexamplex environment:  ∫  eˆx dx   !#@$^&\
The earlier examples had three unmatched open-braces which had to be
closed in this environment.  Also the backslash had to be doubled.
Percent signs get ignored---a BUG.
```

Warning: *The `codeexamplex` environment ideally should ignore `%`s in the first column and print all others. The best I can do for the moment is to ignore all `%`s.*

`bothexample`
`bothexample*`

The `bothexample` and `bothexample*` environments typeset their contents two times, side by side. The left side is set like either a `codeexample` or `codeexample*` environment, and the right side is set like a `typesetexample` environment.

I find setting a line width of 65 characters in your editor helps keep this kind of example from bleeding to the right.

```
This is one part of a \env{bothexample} environment:
\(\int e^x\,dx\)  \fbox{jub-jub} \par
The contents must not contain any \LaTeX{} errors.

There is an annoying \% plus three explicit spaces at the beginning of
the first part.  They are a bug!  Also notice also the missing percent
sign after the backslash after ``annoying''---another bug that affects
the verbatim part only.
```

LOOKS LIKE:

This is one part of a `bothexample` environment: $\int e^x\,dx$ $\boxed{\text{jub-jub}}$
The contents must not contain any LaTeX errors.
There is an annoying % plus three explicit spaces at the beginning of the first part. They are a bug! Also notice also the missing percent sign after the backslash after "annoying"—another bug that affects the verbatim part only.

`splitexample`
`splitexample*`

The `splitexample` environment takes an argument which is used on the right side (the typeset side) of an element that looks like `bothexample`. The contents of the environment are set on the left side. This is a way to cheat, making the right side something other than the strict tyepsetting of what's on the left.

Warning: *The `splitexample` and `bothexample` environments execute example code. The example code is executed in a group, but be careful with global assignments, and with assignments that might screw up the example environment itself before the group ends. Notice that `\setcounter` is global.*

The `bothexample` and `splitexample` environments use an auxiliary file with extension `vrb`, but a second pass with LaTeX is not necessary.

To do: *Implement* `splitexample` *and* `splitexample*`*.*

To do: *Get the* `bothexample` *sub-environments side by side! Right now they are sequential. I recommend keeping line length inside this environment less than 40, however, in case the side-by-side presentation is ever possible. I have not followed this advice myself.*

To do: *Implement* `codeexamplex*`*.*

Warning: *The* `bothexample` *environment is working nicely now, but I have not even touched the other complicated verbatim-like environments, so they might be broken.*

`\url` `\email` `\path` `\file`

The *compsci* package uses the *url* package by Donald Arseneau to handle the typesetting of URLs, email addresses, and filesystem paths. See that package for details of how line breaks are handled within these elements.

`\option`

`\option {⟨option name⟩}` typesets the name of a package or class option. For example, the `landscape` option.

`\program`

`\program {⟨program name⟩}` typesets the name of a computer program. For example the `xdvi` program. See also `\newprogram` below.

`\lips`

The *compsci* package uses the *lips* package, so you can use the `\lips` command for text ellipses. My love is like a red red rose. . . .

`\book`

The *compsci* package uses the *titles* package, so several commands like `\book` are available.

2 Referring to commands

`\cs` `\cmd`

`\cs {⟨command sequence⟩}` typesets a command sequence, such as the one that starts this sentence. `\cmd` does the same thing. For example, `\cs\foo` looks like `\foo`. When writing, e.g., `\cs\foo` the following spacing is automatically handled. The result doesn't look different from using the `|...|` or `\code` syntax, but the markup allows the parsing of the macro name itself for some future purpose such as indexing.

Warning: *You cannot say* `\cs\foo@bar` *when* `@` *is not a letter (category code 11), you must say* `\cs{\foo@bar}` *or* `\cname{foo@bar}` *instead.*

Nor can you say `\cs\par`*. It trips on* `\text@command`*.*

Nor can you say `\cs\iffoo`*. Use* `\cname{iffoo}` *instead.*

`\cname`

`\cname {⟨command name⟩}` is like `\cs` but its argument is evaluated and prefixed with a backslash character. For example, writing `\cname{@tfor}` gives you `\@tfor`.

`\marg` `\oarg`

`\marg {⟨mandatory argument⟩}` typesets a mandatory argument, and `\oarg {⟨optional argument⟩}` typesets an optional argument. For example, These macros are defined with `\meta`, so you can use `\\` for a typeset backslash.

```
\cs\GobbleMOM\marg{first arg}\oarg{optional second arg}\marg{third arg}
```

LOOKS LIKE:

`\GobbleMOM {`⟨*first arg*⟩`}[`⟨*optional second arg*⟩`]{`⟨*third arg*⟩`}`

`\meta`

`\meta {⟨text⟩}` refers to a syntactic placeholder. You would use this for example to talk about one of the arguments of a command you've introduced, like ⟨*text*⟩ above, which can contain normal text but is broken across lines only at spaces. Within the argument `\\` is a typeset backslash.

3 Miscellaneous

`\newprogram` The *compsci* package uses the *abbrevs* package to define the `Program` category of abbreviation. Use `\newprogram{`⟨`\`*csname*⟩`}{`⟨*initial expansion*⟩`}[`⟨*subsequent expansion*⟩`]` to define a new abbreviation `{`⟨`\`*csname*⟩`}` for a program.

`\ProcessDTXFile` `\ProcessDTXFile` can be used by a documentation driver file of the kind you might be inclined to make using the *doc* package. Its purpose is to be able to load a package up to a certain point so that just descriptive information can be read without reading in the whole file. `\ProcessDTXFile{`⟨*filename*⟩`}` defines the variable `\JusTLoaDInformatioN` and inputs the file ⟨*filename*⟩. To be useful, the file must have a statement that branches on whether `\JusTLoaDInformatioN` is defined. For an example of its use, see the present package.

`\MaybeBibliography`

Sorry, you will need to look at the source below.

`\AddToCheckSum` `\AddToCheckSum{`⟨*number*⟩`}` simply adds ⟨*number*⟩ globally to the checksum counter implemented in the *doc* package. This macro finds applications when the sources of a package are spread over one or more files such as configuration files or files concatenated by a `Make` process. For an example of its use, see the present package.

`\BibTeX` The BibTeX logo is provided if it isn't already defined.

Part II

Implementation

4 Version control

`\fileinfo` `\DoXUsepackagE` `\HaveECitationS` `\fileversion` `\filedate` `\docdate` `\PPOptArg`

These definitions must be the first ones in the file.

```
\def\fileinfo{macros for writing about macros (Frankenstein's conscience)}
\def\DoXPackageS {}
\def\fileversion{v1.2}
\def\filedate{2001/08/31}
\def\docdate{2001/08/31}
\edef\PPOptArg {%
  \filedate\space \fileversion\space \fileinfo
}
```

If we're loading this file from a `\ProcessDTXFile` command (see the *compsci* package), then `\JusTLoaDInformatioN` will be defined; othewise we assume it is not (that's why the FunkY NamE).

If we're loading from `\ProcessDTXFile`, we want to load the packages listed in `\DoXPackageS` (needed to typeset the documentation for this file) and then bail out. Otherwise, we're using this file in a normal way as a package, so do nothing. `\DoXPackageS`, if there are any, are declared in the `dtx` file, and, if you're reading the typeset documentation of this package, would appear just above. (It's OK to call `\usepackage` with an empty argument or `\relax`, by the way.)

```
\makeatletter% A special comment to help create bst files.  Don't change!
\@ifundefined{JusTLoaDInformatioN} {%
  }{% ELSE (we know the compsci package is already loaded, too)
  \UndefineCS\JusTLoaDInformatioN
  \SaveDoXVarS
  \eExpand\csname DoXPackageS\endcsname\In {%use \csname in case it's undefined
    \usepackage{#1}%
  }%
  \RestoreDoXVarS
  \makeatother
  \endinput
}% A special comment to help create bst files.  Don't change!
```

Now we check for LaTeX2e and declare the LaTeX package.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{compsci}[\PPOptArg]
```

5 Requirements and context

```
\RequirePackage{abbrevs,alltt,lips,moredefs,relsize,shortvrb,%
                slemph,titles,url,verbatim}
```

6 Hyperlatex option

Handle `hyperlatex` option here.

To do: *Should this inherit globally?*

Long URLs are so common and screw up LaTeX in the middle of a paragraph that a good scheme seems to be to relegate them to footnotes. Make option/switch for footnotes and inline.

Actually, when it's a footnote, there's almost never going to be a linebreak, unless the url is longer than the textwidth of the footnote...so `\textt{}` is probably going to be fine.

Could also use `\marginpar{\smaller\url{#1}}`—definitely would have to use `\url` not `\textt` there.

We undefine `\code` and `\file` immediately because we want *compsci*'s versions not *hyperlatex*'s.

```
\DeclareOption{hyperlatex} {%
 \FrankenInfo{compsci}{hyperlatex}
 \UndefineCS\code
 \UndefineCS\file
 \AtEndOfPackage {
%  \UndefineCS\code  hunh?????
  \DeleteShortVerb{\|}
  \newlet\cs@url\url
  \renewcommand*\url [1][] {%
   \def\sc@t@a{#1}%
   \ifx\sc@t@a\ShortEmpty
    \let\sc@t@a\cs@url
   \else
    \defcommand\sc@t@a [1] {%
     #1\footnote{See \texttt{##1}.}%
    }%
   \fi
   \sc@t@a
  }
  \renewcommand*\email [1][] {%
   \def\sc@t@a{#1}%
   \ifx\sc@t@a\ShortEmpty
    \expandafter\cs@email
   \else
    #1 \expandafter\cs@email
   \fi
  }
 }
}
\ProcessOptions
```

To do: *Alternative ways to handle things*

```
%%
%  \def\sc@t@a {%
%    #1%
%    \begingroup
%     \def\UrlLeft {( see}%
%     \def\UrlRight {)}%
%     \Url
%   }%
%%
%  \def\sc@t@a {%
%    #1%
%       \begingroup
%         \def\UrlLeft{\footnote\bgroup}%
%         \def\UrlRight {.\egroup}%
%         See \Url
%     }%
```

7 Textual elements

`\code`
`|...|`
`\url@ttnobreakstyle`

We use the *url* package to implement `\code` and the *shortvrb* package to implement `|...|`. There are no linebreaks in either. See the respective packages for considerations of robustness.

To do: *summarize robustness considerations here.*

```
\MakeShortVerb{\|}
\newcommand*\code {%
  \begingroup
    \urlstyle{ttnobreak}%
    \Url
}
\newcommand\url@ttnobreakstyle {%
  \def\UrlFont     {\ttfamily}%
  \def\UrlSpecials {\do\ {\ }}%
  \def\UrlOrds     {\do\*\do\-}%
}
```

`codeexample`
`codeexample*`

What I'm doing here is defining `codeexample` and `codeexample*` to be just like the *doc* package's `verbatim` and `verbatim*`. That is, they ignore `%` characters at the beginning of lines.

```
\begingroup
  \catcode `|=0
  \catcode `[=1
  \catcode `]=2
  \catcode`\{=12
  \catcode`\}=12
  \catcode`\\=12
  |gdef|@xverbatim#1\end{codeexample}[#1|end[codeexample]]
  |gdef|@sxverbatim#1\end{codeexample*}[#1|end[codeexample*]]
|endgroup
\newcommand\codeexample {%
  \@beginparpenalty
  \predisplaypenalty
  \@docverbatim
  \MacroFont
  \frenchspacing
  \@vobeyspaces
  \@xverbatim
}
\NewName{codeexample*} {} {%
  \@beginparpenalty
  \predisplaypenalty
  \@docverbatim
  \MacroFont
  \@sxverbatim
}
\newcommand\@docverbatim {%
  \trivlist
  \item[]%
  \if@minipage\else
    \vskip\parskip
  \fi
  \leftskip\@totalleftmargin
  \rightskip\z@
  \parindent\z@
  \parfillskip\@flushglue
  \parskip\z@
  \@@par
  \@tempswafalse
  \def\par {%
    \if@tempswa
      \hbox{}%
    \fi
    \@tempswatrue
    \@@par
    \penalty\interlinepenalty
    \check@percent
  }%
  \obeylines
  \let\do\do@noligs
  \verbatim@nolig@list
  \let\do\@makeother
  \dospecials
}
```

`\MakePercentIgnore`
`\MakePercentComment`

These are defined in the *doc* package, which may be loaded.

```
\requirecommand*\MakePercentIgnore{\catcode`\%9\relax}
\requirecommand*\MakePercentComment{\catcode`\%14\relax}
```

`codeexamplex`

The `codeexamplex` environment ideally should ignore `%`s in the first column and print all others. The best I can do for the moment is to ignore `%`s.

To do: *use verbatim-write trick*

```
\newenvironment{codeexamplex} {%
    \alltt
    \def\\{\char`\\}%
    \MakePercentIgnore\relax
  }{%
  \endalltt
}
```

`typesetexample`

The `typesetexample` environment is simple, it's just the `quote` environment.

```
\newenvironment{typesetexample} {%
    \quote
  }{%
  \endquote
}
```

To implement the `bothexample` environments we write out the contents of the environment and read them in twice, typesetting them differently each time. The `verbatimwrite` environment comes from the *moreverb* package. Reading the text twice seems unavoidable because we want to read them with different catcodes each time.

To do: *Figure whether the tab stuff here is going is a help or a hindrance.*

`\verbatim@out`
`\end@cs@verbatim@write`
`\cs@verbatim@write`
`\verbtab@oktrue`
`\verbtab@okfalse`
`\ifverbtab@ok`
`\if@cs@first@line@`
`\@cs@first@line@true`
`\@cs@first@line@false`
`\cs@verbatim@input`

```
\newwrite \verbatim@out
\DefName{end@cs@verbatim@write} {} {%
  \immediate\closeout\verbatim@out
  \@esphack
}
\defcommand\cs@verbatim@write [1] {%
  \@bsphack
  \immediate\openout \verbatim@out #1
  \let\do\@makeother\dospecials
  \catcode‘\^^M\active \catcode‘\^^I=12
  \def\verbatim@processline {%
    \if@cs@using@doc@
      \immediate\write\verbatim@out{\expandafter\GobbleM\the\verbatim@line}%
    \else
      \immediate\write\verbatim@out{\the\verbatim@line}%
    \fi
  }%
  \verbatim@start
}
\newboolean{verbtab@ok}
\newboolean{@cs@first@line@}
\ReserveCS\cs@verbatim@input
\@cs@first@line@true
{\catcode‘\~=\active \lccode‘\~=‘\^^M \lccode‘\N=‘\N
  \lowercase{%
    \gdef\cs@verbatim@input #1{%
      \begingroup
        \@verbatim\frenchspacing\@vobeyspaces
        \def\verbatim@addtoline##1~{%
          \verbatim@line\expandafter{\the\verbatim@line##1}%
        }%
        \openin\verbatim@in@stream=#1
        \ifeof\verbatim@in@stream
          \FrankenError{compsci}
            {Can’t find file #1!}\@eha
        \else
          \verbtab@oktrue
          \verbatim@line {}%
          \def\verbtab@prev@line{PREVINIT~}%
          \loop
            \read\verbatim@in@stream to \verbtab@line
            \ifeof\verbatim@in@stream
              \verbtab@okfalse
            \else
%              \DTypeout{My line is: [\meaning\verbtab@line]}%
              \if@cs@first@line@
%                \DTypeout{I’m avoiding printing: [\meaning\verbtab@prev@line]}%
                \@cs@first@line@false
              \else
%                \DTypeout{I'll print: [\meaning\verbtab@prev@line]}%
                \expandafter\verbatim@addtoline\verbtab@prev@line
                \the\verbatim@line
                \par
                \verbatim@line {}%
              \fi
              \let\verbtab@prev@line\verbtab@line
            \fi
          \ifverbtab@ok \repeat
          \closein\verbatim@in@stream
        \fi
        \endtrivlist
      \endgroup
      \@doendpe
    }%               \gdef
  }%                \lowercase
}%                  catcodes
```

`bothexample`
`bothexample*`

The first `\begingroup` . . . `\endgroup` I think is necessary to contain the verbatim activity. The second is necessary so that activity in the example environment does not effect anything outside it. Of course global activity will, so be careful! The `\minipage` effects the second group. [Hmm, I don't see a "second" group. . . .]

```
\newboolean{@cs@using@doc@}
\newenvironment{bothexample} {%
    \@ifundefined{ver@doc.sty} {%
        \@cs@using@doc@false
      }{% ELSE
      \@cs@using@doc@true
    }
    \par
    \begingroup
     \cs@verbatim@write{\jobname.vrb}
  }{%
  \end@cs@verbatim@write
  \cs@verbatim@input{\jobname.vrb}%
  \endgroup
  \par\noindent{\sffamily\larger LOOKS LIKE:}%
  \par\smallskip
  \noindent\fbox{%
    \minipage{\textwidth}
      \MakePercentComment
      \input{\jobname.vrb}%
    \endminipage
  }%
  \par\medskip
}
```

`splitexample`
`splitexample*`

We can't use `#1` in the end-environment part, so we write it out. Again we have the problem of `%`s.

Warning: *This is broken!*

```
% \newenvironment{gogog}[1]
%   {\verbatimwrite{\jobname.vrb}#1}
%   {\endverbatimwrite}
\newenvironment{splitexample} [1] {%
  \cs@verbatim@write{\jobname.vrb}
    #1
  \end@cs@verbatim@write
  \codeexample
  }{%
  \endcodeexample
  \par\smallskip
  \noindent\fbox{%
    \minipage{\textwidth}
    \input{\jobname.vrb}%
    \endminipage
  }%
  \par\medskip
}
```

\caveat
warning
\cs@caveat
\todo
todoenv
\cs@todo

```
\newcommand\caveat [1] {%
  \cs@caveat
  #1%
  \endgroup
  \par
}
\newenvironment{warning} {%
    \cs@caveat
  }{%
  \endgroup
  \par
}
\newcommand\cs@caveat {%
  \par
  \begingroup
    \em {\bfseries Warning:}\space
}
\newcommand\todo [1] {%
  \cs@todo
    #1%
  \endgroup
  \par
}
\newenvironment{todoenv} {%
    \cs@todo
  }{%
  \endgroup
  \par
}
\newcommand\cs@todo {%
  \par
  \begingroup
    \em {\bfseries To do:}\space
}
```

\program
\bst
\class
\env
\ext
\file
\option
\package
\cat
\typeset

Abstract markup of terms and objects.

```
\NewTextFontCommand \program \TMFontProgram % generic name of computer program
\NewTextFontCommand \bst     \ttfamily      % name of BibTeX bibliography style
\NewTextFontCommand \class   \slshape       % name of LaTeX class
\NewTextFontCommand \env     \ttfamily      % name of environment
\NewTextFontCommand \ext     \ttfamily      % name of file extension
\NewTextFontCommand \option  \ttfamily      % name of package or class option
\NewTextFontCommand \package \slshape       % name of LaTeX package
\NewTextFontCommand \cat     \sffamily      % name of a category
\newlet             \typeset \fbox          % text as it appears when typeset
\newlet             \file    \path          % name of disk file
```

`\email` `\cs@email@nobrace` `\cs@email`

The `hyperlatex` option will change `\email` at the end of this package.

```
\newcommand\cs@email {%
  \begingroup
    \def\UrlLeft{<}%
    \def\UrlRight{>}%
    \urlstyle{tt}%
    \Url
}
\newcommand\cs@email@nobrace {%
  \begingroup
%    \def\UrlLeft{<}%
%    \def\UrlRight{>}%
    \urlstyle{tt}%
    \Url
}
\newcommand\email{\cs@email}
```

8 Referring to commands

`\meta` `\m@ta` `\do@space` `\sp@ce` `\sp@@ce`

Here I redefine `\meta` to include a `\normalfont` right at the beginning inside a group, since the command was working strangely when `\ttfamily` was in effect. Also, I redefine `\\` to be a typeset backslash.

```
\begingroup
\obeyspaces%
\catcode`\^^M\active%
\gdef\meta{\begingroup\let\\\textbackslash\normalfont\obeyspaces\catcode`\^^M\active%
\let^^M\do@space\let \do@space%
\def\-{\egroup\discretionary{-}{}{}\hbox\bgroup\itshape}%
\m@ta}%
\endgroup
```

These are simply from the *doc* package.

```
\defcommand\m@ta [1] {%
  \leavevmode
  \hbox\bgroup$\langle$\itshape#1\/$\rangle$\egroup
  \endgroup
}
\def\do@space {%
  \egroup
  \space
  \hbox\bgroup\itshape\futurelet\next\sp@ce
}
```

```
\def\sp@ce {%
  \ifx\next\do@space
    \expandafter\sp@@ce
  \fi
}
\defcommand\sp@@ce [1] {%
  \futurelet\next\sp@ce
}
```

`\marg` `\oarg`

These use `\meta` and are therefore slightly different than their counterparts in *ltxdoc*. A tiny bit of breakability is added at the beginning.

```
\defcommand*\marg [1] {%
  \penalty 9999%
  {\ttfamily\char`\{\meta{#1}\char`\}}%
}
\defcommand*\oarg [1] {%
  \penalty 9999%
  {\ttfamily[\meta{#1}]}%
}
```

`\cname` `\cmd` `\cs`

For referring to a LaTeX macro. The code is a mix from the *ltxdoc* class and the *abbrevs* package. There is no need to check for `\nocorr` commands inside the argument so we skip `\text@command`. We might conceivably be in a typewriter slanted, so we do check for italic corrections. We don't handle the case of calling this in math mode (see the definition of `\DeclareTextFontCommand` in the kernel).

We handle the cases of `\cmd{\foo}` and `\foo` differently. The latter needs logic to determine the following space, whereas the former does not. They both need italic corrections.

```
\NewRobustCommand*\cname [1] {%
  \expandafter\code\expandafter{\bslash#1}%
}
\DeclareRobustCommand*\cmd {%
  \leavevmode
  \@ifnextchar \bgroup
      \cs@cmd@grouped
    \cs@cmd@ungrouped
}
\newcommand\cs@cmd@grouped [1] {%
  \begingroup
    \maybe@ic
    \code{#1}%
    \aftergroup\maybe@ic
  \endgroup
}
\newcommand\cs@cmd@ungrouped [1] {%
  \begingroup
    \maybe@ic
    \code{#1}%
    \aftergroup\maybe@ic@space
  \endgroup
}
\let\cs\cmd
```

9 Miscellaneous

\partname
\ProcessDTXFile
\IfJustLoadInformation
\JusTLoaDInformatioN
\IfCitations
\initelyHavECitationS
\SaveDoXVarS
\RestoreDoXVarS
\RestorECitationS
\AddToCheckSum

See the version control section above and all throughout the `Frankenstein` bundle for examples of when you might want to use these commands.

```
\def\partname {%
  Part%
}
```

We make sure it's OK to use `\def` multiple times. You might need to call packages with options.

```
\newcommand*\ProcessDTXFile [1] {%
  \def\filename {#1}%
  \def\JusTLoaDInformatioN {}%
  \UndefineCS\initelyHavECitationS
  \input{#1}%
  \UndefineCS\JusTLoaDInformatioN % not necessary in my usage, but safe
}
\newcommand*\IfCitations [1] {%
  \@ifundefined{initelyHavECitationS} {%
    }{% ELSE
    #1\relax
  }%
}
\newcommand*\IfJustLoadInformation [1] {%
  \@ifundefined{JusTLoaDInformatioN} {%
    }{% ELSE
    #1\relax
  }%
}
\newcommand*\SaveDoXVarS {%
  \SaveCS\fileinfo \SaveCS\fileversion
  \SaveCS\filedate \SaveCS\docdate
  \SaveCS\DoXPackageS
  \def\RestorECitationS {%
    \UndefineCS\initelyHavECitationS
  }%
  \IfCitations {%
    \let\RestorECitationS\relax
  }%
}
\newcommand*\RestoreDoXVarS {%
  \RestoreCS\fileinfo \RestoreCS\fileversion
  \RestoreCS\filedate \RestoreCS\docdate
  \RestoreCS\DoXPackageS
  \RestorECitationS
}
\newcommand*\AddToCheckSum [1] {%
  \global\advance\check@sum #1%
}
```

\newprogram
\TMFontProgram
\TMHookProgram
\TMResetProgram

Provide for program abbreviations.

```
\NewAbbrevCategory{Program}
\NewUserAbbrevDefiner{\newprogram}{Program}
\def\TMFontProgram {%
```

```
  \relsize{-1}\ttfamily
}
```

`\BibTeX` The BibTeX logo.

```
\providecommand\BibTeX {%
  {\rmfamily B\kern-.05em{\scshape i\kern-.025em b}\kern-.08em
    T\kern-.1667em\raisebox{-.7ex}{E}\kern-.125emX}%
}
```

Part III

Configuration

Other things are deferred to a configuration file.

```
\InputIfFileExists{compsci.cfg}{}{}
```

The contents of the distributed configuration file are below.

```
\def\fileinfo{Compsci package configuration}
\def\fileversion{v1.2}
\def\filedate{2001/08/31}
\def\docdate{2001/08/31}
```

```
\ProvidesFile{compsci.cfg}
```

`\ALaTeX`
`\Frankenstein`
`\monster`
`\CTAN`
`\kpse`
`\gemacs`
`\auctex`
`\nts`
`\MakeIndex`
`\etex`
`\LaTeXiii`
`\idvi`

Various program names.

```
\newprogram\Frankenstein{Frankenstein}
\newlet\monster\Frankenstein
```

To do: *Hmm, ALaTeX kernel uses* `\DeclareRobustCommand`. *What does LaTeX kernel do?*

```
%\requirecommand\ALaTeX {%
\defcommand\ALaTeX {%
  A\LaTeX%
}
\newabbrev\ctan{CTAN}
\newprogram\kpse{kpathsea}
\newprogram\gemacs{\textrm{GNU} Emacs}
\newprogram\auctex{AUC-\TeX}
\newprogram\nts{$\cal NTS$}
\newprogram\MakeIndex{{\normalshape\em Makeindex\/}}
\newprogram\etex{e-\TeX}
\newprogram\LaTeXiii{\LaTeX~3}
\newprogram\idvi{IDVI}
```

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
\% 136, 137
* 80
\- 80, 322
\/ 18, 327
\@@par 119, 126
\@beginparpenalty 93, 102
\@bsphack 156
\@cs@first@line@false 150
\@cs@first@line@true 150
\@cs@using@doc@false 219
\@cs@using@doc@true 221
\@docverbatim 95, 104, 108
\@doendpe 212
\@eha 184
\@esphack 153
\@flushglue 117
\@ifnextchar 356
\@ifundefined 10, 218, 386, 392
\@makeother 133, 158
\@sxverbatim 106
\@tempswafalse 120
\@tempswatrue 125
\@totalleftmargin 114
\@verbatim 177
\@vobeyspaces 98, 177
\@xverbatim 99
\\ 88, 140, 320
\{ 86, 345
\} 87, 345
\^ 159, 173, 319, 320
\| 31, 71
\~ 173
\␣ 79

A
\active 159, 173, 319, 320
\AddToCheckSum *5*, 375
\advance 415
\aftergroup 364, 371
\ALaTeX *7*
\alltt 139
\AtEndOfPackage 29
\auctex *7*

B
\begingroup 58, 66, 73, 82, 176, 224, 272, 289, 303, 310, 317, 320, 361, 368
\bfseries 273, 290
\bgroup 67, 322, 327, 333, 356
\BibTeX *5*, 422
\book *4*
bothexample (environment) *3*, 216
bothexample* (environment) *3*, 216
\bslash 352
\bst *2*, 292

C
\cal 17
\cat 292
\catcode 83–88, 136, 137, 159, 173, 319, 320
\caveat *2*, 258
\char 140, 345
\check@percent 128
\check@sum 415
\class *2*, 292
\closein 208
\closeout 152
\cmd *4*, 351
\cname *4*, 351
\code *2*, 27, 30, 71, 352, 363, 370
\code* *2*
\codeexample 92, 247
codeexample (environment) *2*, 82
codeexample* (environment) *2*, 82
codeexamplex (environment) *2*, 138
codeexamplex* (environment) *2*
\cs *4*, 351
\cs@caveat 258
\cs@cmd@grouped 357, 360
\cs@cmd@ungrouped 358, 367
\cs@email 47, 49, 302
\cs@email@nobrace 302
\cs@todo 258
\cs@url 32, 36
\cs@verbatim@input 150, 228
\cs@verbatim@write 150, 225, 244
\csname 14
\CTAN *7*
\ctan 13

D
\DeclareOption 25
\DeclareRobustCommand 354
\def 1–5, 34, 45, 56, 59, 60, 64, 67, 68, 78–80, 121, 140, 160, 178, 188, 304, 305, 311, 312, 322, 330, 335, 375, 379, 380, 401, 419
\defcommand 10, 38, 155, 325, 340, 343, 347
\DefName 151
\DeleteShortVerb 31
\discretionary 322
\do 79, 80, 131, 133, 158
\do@noligs 131
\do@space 317
\docdate 1, 5, 399, 410
\dospecials 134, 158
\DoXPackageS 2, 400, 411
\DoXUsepackagE 1
\DTypeout 194, 196, 199

E
\edef 6

\eExpand 14
\egroup 68, 322, 327, 331
\else 37, 48, 111, 163, 185, 193, 198
\em 18, 273, 290
\email *4*, 44, 302
\end 89, 90
\end@cs@verbatim@write 150, 227, 246
\endalltt 143
\endcodeexample 249
\endcsname 14
\endgroup 211, 229, 261, 267, 278, 284, 324, 328, 365, 372
\endinput 19
\endminipage 236, 254
\endquote 148
\endtrivlist 210
\endverbatimwrite 242
\env *2*, 292
environments:
  bothexample* *3*, 216
  bothexample *3*, 216
  codeexample* *2*, 82
  codeexamplex* *2*
  codeexamplex *2*, 138
  codeexample *2*, 82
  splitexample* *3*, 240
  splitexample *3*, 240
  todoenv *2*, 258
  typesetexample *2*, 145
  warning *2*, 258
\etex *7*
\expandafter 47, 49, 162, 179, 200, 337, 352
\ext *2*, 292

F
\fbox 232, 251, 300
\fi 41, 50, 113, 124, 165, 204, 206, 209, 338
\file *2*, *4*, 28, 292
\filedate 1, 4, 399, 410
\fileinfo 1, 2, 398, 409
\filename 379
\fileversion 1, 3, 398, 409
\footnote 39, 67
\FrankenError 183
\FrankenInfo 26
\Frankenstein *7*
\frenchspacing 97, 177
\futurelet 333, 341

G
\gdef 175, 213, 320
\gemacs *7*
\global 415
\GobbleM 162

H
\HaveECitationS 1
\hbox 123, 322, 327, 333

I
\idvi *7*
\if@cs@first@line@ 150
\if@cs@using@doc@ 161
\if@minipage 111
\if@tempswa 122
\IfCitations 375
\ifeof 182, 191
\IfJustLoadInformation 375
\ifverbtab@ok 150
\ifx 35, 46, 336
\immediate 152, 157, 162, 164
\In 14
\initelyHavECitationS 375
\input 235, 253, 382
\InputIfFileExists 1
\interlinepenalty 127
\item 110
\itshape 322, 327, 333

J
\jobname 225, 228, 235, 241, 244, 253
\JusTLoaDInformatioN 12, 375

K
\kern 423, 424
\kpse *7*

L
\langle 327
\larger 230
\LaTeX 11, 20
\LaTeXiii *7*
\lccode 173
\leavevmode 326, 355
\leftskip 114
\let 36, 131, 133, 158, 205, 320, 321, 374, 405
\lips *4*
\loop 189
\lowercase 174, 214

M
\m@ta 317
\MacroFont 96, 105
\makeatletter 9
\makeatother 18
\MakeIndex *7*
\MakePercentComment 136, 234
\MakePercentIgnore 136, 141
\MakeShortVerb 71
\marg *4*, 343
\maybe@ic 362, 364, 369
\maybe@ic@space 371
\MaybeBibliography *5*
\meaning 194, 196, 199
\medskip 238, 256
\meta *4*, 317, 345, 349
\minipage 233, 252
\monster *7*

N
\N 173
\NeedsTeXFormat 21
\newabbrev 13
\NewAbbrevCategory 417
\newboolean 169, 170, 216
\newcommand 72, 77, 92, 108, 258, 270, 275, 287, 302, 309, 316, 360, 367, 378, 385, 391, 397, 408, 414
\newenvironment 138, 145, 217, 240, 243, 264, 281
\newlet 8, 32, 300, 301
\NewName 101
\newprogram *5*, 7, 14–21, 417
\NewRobustCommand 351
\NewTextFontCommand 292–299
\NewUserAbbrevDefiner 418
\newwrite 150
\next 333, 336, 341
\noindent 230, 232, 251

\normalfont 320
\normalshape 18
\nts 7

O

\oarg *4*, 343
\obeylines 130
\obeyspaces . . . 318, 320
\openin 181
\openout 157
\option *4*, 292

P

\package *2*, 292
\par 121, 202, 223, 230, 231, 238, 250, 256, 262, 268, 271, 279, 285, 288
\parfillskip 117
\parindent 116
\parskip 112, 118
\partname 375
\path *4*, 301
\penalty . . 127, 344, 348
\PPOptArg 1, 22
\predisplaypenalty 94, 103
\ProcessDTXFile . *5*, 375
\ProcessOptions . . . 54
\program *4*, 292
\providecommand . . . 422
\ProvidesFile 6
\ProvidesPackage . . 22

Q

\quote 146

R

\raisebox 424
\rangle 327
\read 190
\relax 136, 137, 141, 388, 394, 405
\relsize 420
\renewcommand . . . 33, 44
\repeat 207
\requirecommand 9, 136, 137
\RequirePackage . . . 23
\ReserveCS 171
\RestorECitationS . 375
\RestoreCS . . . 409–411
\RestoreDoXVarS 17, 375
\rightskip 115
\rmfamily 423

S

\SaveCS 398–400
\SaveDoXVarS . . . 13, 375
\sc@t@a . . 34–36, 38, 42, 45, 46, 56, 64
\scshape 423
\sffamily 230, 299
\ShortEmpty 35, 46
\slshape 294, 298
\smallskip . . . 231, 250
\sp@@ce 317
\sp@ce 317
\space . . 7, 273, 290, 332
splitexample (environment) *3*, 240
splitexample* (environment) . *3*, 240

T

\TeX 16, 19
\textbackslash 320
\textrm 15
\texttt 39
\textwidth . . . 233, 252
\the . . 162, 164, 179, 201
\TMFontProgram 292, 417
\TMHookProgram 417
\TMResetProgram . . . 417
\todo *2*, 258
todoenv (environment) *2*, 258
\trivlist 109
\ttfamily 78, 293, 295–297, 345, 349, 420
\typeset *2*, 292
typesetexample (environment) . *2*, 145

U

\UndefineCS . . 12, 27, 28, 30, 381, 383, 402
\Url . 61, 69, 75, 307, 314
\url *4*, 32, 33
\url@ttnobreakstyle 71
\UrlFont 78
\UrlLeft 59, 67, 304, 311
\UrlOrds 80
\UrlRight 60, 68, 305, 312
\UrlSpecials 79
\urlstyle . 74, 306, 313
\usepackage 15

V

\verbatim@addtoline 178, 200
\verbatim@in@stream 181, 182, 190, 191, 208
\verbatim@line 162, 164, 179, 187, 201, 203
\verbatim@nolig@list 132
\verbatim@out 150
\verbatim@processline 160
\verbatim@start . . . 167
\verbatimwrite 241
\verbtab@line 190, 194, 205
\verbtab@okfalse . . 150
\verbtab@oktrue . . . 150
\verbtab@prev@line 188, 196, 199, 200, 205
\vskip 112

W

warning (environment) *2*, 258
\write 162, 164

Z

\z@ 115, 116, 118