

The L^AT_EX dtxdescribe Package

v1.03 — 2022/02/01

© 2016–2022 Brian Dunn
bd@BDTechConcepts.com

Describe additional object types in dtx source files.

Abstract

The `doc` package includes tools for describing macros and environments in L^AT_EX source `.dtx` format. The `dtxdescribe` package adds additional tools for describing booleans, lengths, counters, hooks, keys, packages, classes, options, files, commands, arguments, and other objects. `dtxdescribe` also works with the regular document classes, for those who do not wish to use the `ltxdoc` class and `.dtx` files.

Each described item is given a margin tag similar to `\DescribeEnv`, and is listed in the index by itself and also by category. Each item may be sorted further by an optional class. All index entries except code lines are hyperlinked.

The `dtxexample` environment is provided for typesetting example code and its results. Contents are displayed verbatim along with a caption and cross-referencing. They are then `\input` and executed, and the result is shown.

Environments are also provided for displaying verbatim or formatted source code, user-interface displays, and sidebars with titles.

Macros are provided for formatting the names of inline L^AT_EX objects such as packages and booleans, as well as program and file names, file types, internet objects, the names of certain programs, a number of logos, and inline dashes and slashes.

`dtxdescribe` works with the `ltxdoc` class, but also works with the standard classes as well, except that the `macro` and `environment` environments are not supported. Either `makeidx` or `splitidx` may be loaded by the user. `makeidx` will be used by default.

`dtxdescribe` works with pdfT_EX, X_ET_EX, and LuaT_EX, and perhaps other engines as well.

Contents

1	Introduction	6
2	Using dtxdescribe	7
3	The macros, and the dtxexample environment	8
3.1	Macros and environments	8
3.2	Arguments	8
3.3	Booleans, lengths, counters, hooks, keys	9
3.4	Packages, classes, options	9
3.5	Files, programs, commands	10
3.6	Other source objects	10
3.7	In a description environment	10
3.8	Defaults	11
3.9	Nesting	11
3.10	\margin tag, \watchout	12
3.11	dtxexample environment	12
3.12	noindmacro and noindenvironment environments	13
3.13	sourceverb, sourcedisplay, UIdisplay, docsidebar	13
3.14	Formatted objects	14
3.14.1	L ^A T _E X objects	14
3.14.2	Programs and commands	15
3.14.3	File types	15
3.14.4	Internet	16
3.14.5	Specific programs	16
3.14.6	Acronyms, brand names, trademarks	16
3.15	Logos	17

3.16 Dashes and slashes	17
4 Examples	19
5 Usage notes	32
6 Code	33
6.1 Required packages	33
6.2 Gobbling comment characters	35
6.3 Vertical spacing	36
6.4 ltxdoc emulation	36
6.5 doc emulation	36
6.6 Support macros	37
6.7 \DescribeMacro and \DescribeEnvironment	42
6.8 New \Describe. . . macros	43
6.9 \DescribeDefault	48
6.10 \ItemDescribeMacro, etc.	48
6.11 \margintag, \watchout	52
6.12 Nesting	53
6.13 The dtxexample environment	53
6.14 noindmacro and noindenvironment	56
6.15 sourcedisplay, UIdisplay, docsidebar	56
6.16 Formatted objects	58
6.16.1 L ^A T _E X objects	58
6.16.2 Programs and commands	60
6.16.3 File types	60
6.16.4 Internet	61
6.16.5 Specific programs	62

6.16.6 Acronyms, brand names, trademarks	63
6.17 Logos	63
6.18 Dashes and slashes	65
7 Compiling dtxdescribe	67
 Change History and Index	 68

List of Examples

1 Macros	19
2 Environment	20
3 Second Environment	20
4 Booleans and Counters	21
5 Lengths	21
6 Packages, Classes, and Options	22
7 Files, Commands, and Programs	22
8 Keys	23
9 Arguments	24
10 Object	25
11 Other	25
12 Description environments	26
13 Nesting	27
14 dtxexample	28
15 fsourceverb	29
16 sourcedisplay	29
17 UIdisplay	30

dtxdescribe	5
-------------	---

18 docsidebar	31
-------------------------	----

List of Figures

1 A Figure	28
----------------------	----

1 Introduction

The `doc` package provides `\DescribeMacro` and `\DescribeEnv` to help document new macros and environments. Each generates a heading in the documentation, to which `\marg`, `\oarg`, and `\parg` may be added to identify arguments to be passed to the new object. Their names are added to the margin, and index entries are added, as well as group of entries for environments.

`dtxdescribe` extends this concept to include a number of additional objects, such as booleans and keys. To help identify what is being described in the margin, small tags are added to the name, such as “Env”, “Bool”, or “Key”. These new objects are also listed in the index with the same tag shown after their names, and also by group. Optional classes may be used to further categories index entries.

Modifications have been made to interact with `hyperref` to provide hyper links for regular index entries as well as the new `\Describe` entries.

Additional macros are provided to generate colored margin tags and warnings, and a new `dtxexample` environment demonstrates code examples.

This documentation and its index show examples of these macros in use.

While the index may appear to be overkill for a small package, keep in mind that it includes a number of fictional entries from the examples. Extensive cross-referencing can be useful for larger works. And, of course, you need not cross-reference everything!

2 Using dtxdescribe

Place `\usepackage{dtxdescribe}` in the .dtx file's driver section:


```
%<*driver>
\documentclass{ltxdoc}
...
\usepackage{lmodern}
...
\usepackage{dtxdescribe}
...
\usepackage{packagename} % the name of your new package
...
\usepackage[...]{hyperref}
\usepackage[...]{cleveref}
...
%</driver>
```

Various objects inside the dtx file may be described with new macros such as `\DescribeBoolean`, `\DescribeLength`, `\DescribeCounter`, similar to the already-familiar `\DescribeMacro` and `\DescribeEnv`.

Optional “classes” may be assigned to the objects being described, including the new versions of `\DescribeMacro` and `\DescribeEnv`. These classes are printed in the margin tag and index entry for each item, and also generate additional index entries sorted by class. This is especially useful for key/value sets, where several sets may appear in the same document.

inside a float The margin tag is not printed if the `\Describe` macros are used inside a float such as a table, but the index entries are still made.

`\margintag{text}` `\margintag{text}` may be used to place a colored tag in the margin to summarize paragraph contents or draw attention to an index destination.

 `\watchout[optional text]` `\watchout[optional text]` may be used to place a red warning sign in the margin, along with optional text.

The `dtxexample` environment may be used to typeset and execute small pieces of L^AT_EX code as examples of its use. Optional cross-referencing notes may be used to refer to any example float being generated.

3 The macros, and the dtxexample environment

3.1 Macros and environments

Env macro

These are only provided by the `ltxdoc` class and `doc` package to document a `.dtx` file, where comments are used by *docstrip* to disable these environments in the resulting `.sty` file. When using the regular document classes, the macro and environment environments would localize any definitions, and `\DescribeMacro` and `\DescribeEnv` should be used instead.

Env environment

 **.dtx only**

`\DescribeMacro`

`[\langle class \rangle] {\langle \name \rangle}`

The preexisting macro from the `doc` package is redefined to create hyperlinked index entries, and include an optional class. A margin tag is created and an index entry is made. When the optional class is used, it is displayed in front of the margin tag, and is used to group an index entry by macro name and another index entry by class. An example would be to describe the float creation and caption setup for a new class of float, such as the `dtxexample` float and the example “photograph” float both found in the index for this document. See example 1 on page 19 for examples.

`\DescribeEnv`

`[\langle class \rangle] {\langle environment name \rangle}`

The preexisting macro from the `doc` package is redefined to create hyperlinked index entries, include an optional class, and also to place an ‘Env’ tag in front of the name in the margin. See example 2 on page 20.

3.2 Arguments

The `\Describe. . .` macros may be followed by `\marg`, `\oarg`, and `\parg` to describe arguments passed to the macros.

`\marg`

`{\langle text \rangle}`

Shows a mandatory argument for a macro or environment.

The results looks like `{\langle mandatory \rangle}`.

`\oarg`

`{\langle text \rangle}`

Shows an optional argument for a macro or environment.

The results looks like `[\langle optional \rangle]`.

`\parg`

`{\langle text \rangle}`

Used for “picture” arguments, such as coordinates.

The result looks like ($\langle coordinate \rangle$).

`\DescribeArgument`

`[$\langle class \rangle$] { $\langle argument \rangle$ }`

May be used to describe actions taken when given certain macro arguments. These will be given an ‘Arg’ margin tag and will appear in the index. The class may be used to categorize arguments by their macro or environment name. See example 9 on page 24.

3.3 Booleans, lengths, counters, hooks, keys

See example 4 on page 21.

`\DescribeBoolean`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a boolean. Given a ‘Bool’ tag in the margin and index.

`\DescribeLength`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a length. Given a ‘Len’ tag in the margin and index.

`\DescribeCounter`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a counter. Given a ‘Ctr’ tag in the margin and index.

`\DescribeHook`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a hook. Given a ‘Hook’ tag in the margin and index. The class may be used to categorize hooks by package. Example:

```
\DescribeHook[LaTeX]{para/begin}
```

`\DescribeKey`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a key. Given a ‘Key’ tag in the margin and index. The class may be used to categorize keys by their kev/value group. See example 8 on page 23.

3.4 Packages, classes, options

`\DescribePackage`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a package. Given a ‘Pkg’ tag in the margin and index.

`\DescribeClass`

`[$\langle class \rangle$] { $\langle name \rangle$ }`

Describes a L^AT_EX class. Given a ‘Cls’ tag in the margin and index.

`\DescribeOption` [`<class>`] {`<name>`}

Describes a L^AT_EX package or class option. Given an ‘Opt’ tag in the margin and index.

3.5 Files, programs, commands

`\DescribeFile` [`<class>`] {`<name>`}

Describes an operating-system file. Given a ‘File’ tag in the margin and index. The filename may have underscores.

`\DescribeProgram` [`<class>`] {`<name>`}

Describes an operating-system program. Given a ‘Prog’ tag in the margin and index. The program name may have underscores.

`\DescribeCommand` [`<class>`] {`<name>`}

Describes an operating-system command. Given a ‘Cmd’ tag in the margin and index. The command name may have underscores.

3.6 Other source objects

`\DescribeObject` [`<class>`] {`<name>`}

Describes an arbitrary programming object, such as a color definition or caption setup. A margin tag and index entry are created with `\ttfamily` type. When a class is used, it is pre-pended to the margin tag, appended to the index entry, and a second index entry is created grouped by class. If a macro name is to be described, use `\DescribeMacro` instead. See example 10 on page 25.

`\DescribeOther` [`<class>`] {`<name>`}

Describes an arbitrary non-programming object, such as a license agreement or credits. A margin tag and index entry are created in roman type. When a class is used, it is pre-pended to the margin tag, appended to the index entry, and a second index entry is created grouped by class. See example 11 on page 25.

3.7 In a description environment

To describe an object using a description environment, use the following. See example 12 on page 26.

<code>\ItemDescribeMacro</code>	<code>[⟨class⟩] {⟨\name⟩}</code> A description.
<code>\ItemDescribeEnv</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeArgument</code>	<code>[⟨class⟩] {⟨argument⟩}</code> A description.
<code>\ItemDescribeBoolean</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeLength</code>	<code>[⟨class⟩] {⟨\name⟩}</code> A description.
<code>\ItemDescribeCounter</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeHook</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeKey</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribePackage</code>	<code>[⟨class⟩] {⟨package_name⟩}</code> With underscores.
<code>\ItemDescribeClass</code>	<code>[⟨class⟩] {⟨class_name⟩}</code> With underscores.
<code>\ItemDescribeOption</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeFile</code>	<code>[⟨class⟩] {⟨file_name⟩}</code> With underscores.
<code>\ItemDescribeProgram</code>	<code>[⟨class⟩] {⟨program_name⟩}</code> With underscores.
<code>\ItemDescribeCommand</code>	<code>[⟨class⟩] {⟨command_name⟩}</code> With underscores.
<code>\ItemDescribeObject</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.
<code>\ItemDescribeOther</code>	<code>[⟨class⟩] {⟨name⟩}</code> A description.

3.8 Defaults

<code>\DescribeDefault</code> Default: value	<code>{⟨value⟩}</code> Shows the default value of a <code>\Describe. . .</code> item, such as displayed here. Place this macro immediately after the <code>\Describe. . .</code> macro and any arguments, but before the text description.
<code>\DescribeDefaultcolor</code> Default: green!50!black	The color of the margin tag used to show the default value. This is used by <code>\textcolor</code> to create the margin tag.

3.9 Nesting


<code>\shownesting</code>	<code>* [⟨fraction of \linewidth⟩] {⟨container name⟩} {⟨contents⟩}</code>
---------------------------	---

It may be useful to show which objects contain which other objects. `\shownesting` shows a box enclosing a name for the container, and the container's contents. `\shownesting` be nested, showing boxes inside other boxes, which displays how each environment and macro is fit together inside each other.

The optional argument is the fraction of `\linewidth` to use for the box, from `[0]` to `[1]`. The default is `[1]`. Each `\shownesting` starts its own paragraph, unless the star `*` is used, in which case the `\shownesting*` appears inline with previous text. To place two `\shownesting` boxes side-by-side, use optional arguments to specify less than full `\linewidth` for each box, and use `\shownesting*` for the second box to place it inline.

See example 13 on page 27 for an example.

3.10 `\margintag`, `\watchout`

<code>\margintag</code>	<code>{\langle text \rangle}</code>
<code>\margintag{example}</code>	Creates a colored margin tag. May be used to identify the topic of a paragraph or the destination of an arbitrary index entry.
<code>\margintagcolor</code> Default: blue!70!black	The color of the <code>\margintag</code> .
<code>\watchout</code>	<code>[\langle text \rangle]</code>
 <code>\watchout{example}</code>	Creates a red margin tag with a warning sign and optional text. May be used to warn the reader of special instructions, etc. Without the optional text the warning sign is displayed by itself.
<code>\watchoutcolor</code> Default: red!50!black	The color of the <code>\watchout</code> .

3.11 `dtxexample` environment

Env <code>dtxexample</code>	<code>* [\langle Notes/cross-references \rangle] {\langle caption & label \rangle}</code>
-----------------------------	---

The `dtxexample` environment is useful for demonstrating a piece of \LaTeX code. The example is a simulated float with its own caption and optional label, along with optional notes and/or cross-referencing commands. The contents of the `dtxexample` environment are printed verbatim, then loaded and executed as \LaTeX code, showing the results just below the printed code. In the case of float commands, the floats are generated as expected somewhere nearby, and should be given their own labels. References to the float's labels may be placed in the optional argument to the `dtxexample` environment, and will be printed below the code.

The unstarred version places the code inside a minipage, forbidding a page break in the middle of the code listing. The starred version does not use a minipage. This is required when the code is too large to fit on a single page.

See example 14 for a demonstration of how dtxexample works.

<code>\dtxexamplecodename</code>	The text name of the code section.
<code>\dtxexampleresultname</code>	The text name of the result section.

Default: Code:

Default: Result:

3.12 noindmacro and noindenvironment environments

⚠ .dtx only These are like macro and environment, but not indexed. These only make sense if using the ltxdoc class and doc package to document a .dtx file, where comments are used by *docstrip* to disable these environments in the resulting .sty file. When using the regular document classes, noindmacro and noindenvironment environments should not be used, as they would localize any definitions. \DescribeMacro and \DescribeEnv should be used instead.

Env noindmacro	{<\name>} To document macros which should not be included in the index.
Env noindenvironment	{<name>} To document environments which should not be included in the index.

Replace

```
\begin{macro}{\macroname} \oarg{optional} \marg{mandatory}
...
\end{macro}
```

with

```
\begin{noindmacro}{\macroname} \oarg{optional} \marg{mandatory}
...
\end{noindmacro}
```

and similarly for noindenvironment.

3.13 sourceverb, sourcedisplay, UIdisplay, docsidebar

Env sourceverb	[<key/values>] Display source code verbatim. Uses optional fancyvrb keys. Includes gobble=2 to absorb the leading % and space character of a dtx file source format. Because this is a verbatim environment, it <i>cannot</i> be used inside a macro.
	Default: gobble=2, tabsize=4, xleftmargin=2em
Env fsourceverb	[<key/values>] Display source code verbatim inside a frame. A label may be
	Default: gobble=2, tabsize=4, xleftmargin=2em, frame=lines

included using the `label` key. Because this is a verbatim environment, it *cannot* be used inside a macro. See example 15 on page 29.

Env <code>sourcedisplay</code>	Display source code with manual formatting. This is not a verbatim environment. <code>\textcolor</code> , <code>\textbf</code> , and <code>\emph</code> may be used to highlight text. Macros must be escaped with <code>\cs</code> , characters such as <code>{</code> must be produced with <code>\{</code> , etc. <code>\\</code> must be used to force a new line. <code>\fquad</code> , <code>\fqquad</code> , and <code>\fqqqquad</code> may be used to force indenting. Because this is <i>not</i> a verbatim environment, it <i>can</i> be used inside a macro. See example 16 on page 29.
<code>\fquad</code>	Single-level indent inside a <code>sourcedisplay</code> .
<code>\fqquad</code>	Double-level indent inside a <code>sourcedisplay</code> .
<code>\fqqqquad</code>	Triple-level indent inside a <code>sourcedisplay</code> .
Env <code>UIDisplay</code>	Displays a user interface, such as a dialog box entry or a menu selection. See example 17 on page 30. Also see the <code>\UI</code> macro..
<code>\userentry</code>	<code>{\langle text to enter \rangle}</code> Typeset something for the user to enter. Also see the <code>\cmds</code> macro.
<code>\userentryname</code> Default: Enter ⇒	Text to tell the user to enter the following item. Change with <code>\renewcommand</code> .
Env <code>docsidebar</code>	<code>[\langle title \rangle]</code> Creates a sidebar within the document. See example 18 on page 31.

3.14 Formatted objects

Macros to format references to various kinds of objects.

This `dtxdescribe` package documentation uses `erewhon`, `cabin`, and `inconsolata`, along with `metalogo`, to demonstrate the following font effects.

3.14.1 L^AT_EX objects

<code>\pkg</code>	<code>{\langle packagename \rangle}</code> Prints as <code>packagename</code> . Also for a <code>classname</code> .
<code>\cs</code>	<code>{\langle csname \rangle}</code> Prints as <code>\csname</code> .
<code>\env</code>	<code>{\langle environment \rangle}</code> Prints as <code>environment</code> .
<code>\marg</code>	<code>{\langle argument \rangle}</code> Prints <code>{\langle arg \rangle}</code> . Mandatory argument.
<code>\oarg</code>	<code>{\langle argument \rangle}</code> Prints <code>[\langle arg \rangle]</code> . Optional argument.

<code>\parg</code>	<code>{\langle argument \rangle}</code> Prints (<code>\langle arg \rangle</code>). Picture-mode argument.
<code>\ctr</code>	<code>{\langle counter \rangle}</code> Prints as counter.
<code>\bool</code>	<code>{\langle boolean \rangle}</code> Prints as boolean.
<code>\optn</code>	<code>{\langle option \rangle}</code> Prints as option, for example to a macro, package, class.
<code>\TOC</code>	toc: Table of contents.
<code>\LOF</code>	LOF: List of figures.
<code>\LOT</code>	LOT: List of tables.

3.14.2 Programs and commands

<code>\progcde</code>	Prints as inline program code: Escape underscores and other special characters such as <code>{</code> , <code>%</code> , <code>\$</code> .
<code>\prog</code>	Prints as <i>grep</i> , <i>make</i> : A program name. Underscores allowed.
<code>\filenm</code>	Prints as <code>file_name</code> : Underscores allowed.
<code>\UI</code>	Prints as General user-interface text . What the user sees on the display. Also see the <code>UIDisplay</code> environment.
<code>\cmds</code>	Prints as Commands to be entered : What the user enters. Escape underscores and other special characters such as <code>{</code> , <code>%</code> , <code>\$</code> . Also see the <code>\userentry</code> macro.

3.14.3 File types

<code>\ODT</code>	ODT OpenDocument Format word processing document
<code>\SVG</code>	SVG image format
<code>\PNG</code>	PNG image format
<code>\GIF</code>	GIF image format
<code>\JPG</code>	JPG image format
<code>\EPS</code>	EPS image format
<code>\PDF</code>	PDF image format
<code>\DVI</code>	DVI image format

3.14.4 Internet

<code>\UTF</code>	UTF: Unicode
<code>\URL</code>	URL: Uniform Resource Locator
<code>\element</code>	$\langle element\ name\rangle$ Prints as <code><element></code> , an HTML/CSS element
<code>\attribute</code>	$\langle attribute\ name\rangle$ Prints as <code>attribute</code> , an HTML/CSS attribute. pdfTeX and XeTeX only. Not for LuaTeX.
<code>\attrib</code>	$\langle attribute\ name\rangle$ Prints as <code>attribute</code> , an HTML/CSS attribute. pdfTeX, XeTeX, or LuaTeX.
<code>\HTML</code>	HTML: Hypertext Markup Language
<code>\HTMLfive</code>	HTML5: Old-style figure if font supports
<code>\CSS</code>	CSS: Cascading Style Sheet
<code>\CSSthree</code>	CSS3: Old-style figure if font supports
<code>\EPUB</code>	EPUB: E-book file format

3.14.5 Specific programs

<code>\tikz</code>	Tikz: Package logo
<code>\CTAN</code>	CTAN: Comprehensive TeX Archive Network
<code>\TDS</code>	TDS: TeX Directory Structure
<code>\MathML</code>	MathML: Mathematical Markup Language
<code>\MathJax</code>	MATHJAX: Math on the web.

3.14.6 Acronyms, brand names, trademarks

<code>\brand</code>	$\langle name\rangle$ BRANDNAME, COMPANY NAME
<code>\acro</code>	$\langle acronym\rangle$ ACRO: Acronym
<code>\supregistered</code>	Superscript trademark symbol®

3.15 Logos

Several additional logos are provided.

Also see the `metalogo` and `metalogoX` packages.

<code>\dviTeX</code>	DVI T _E X
<code>\dviLaTeX</code>	DVI L ^A T _E X
<code>\pdfTeX</code>	PDF T _E X
<code>\pdfLaTeX</code>	PDF L ^A T _E X
<code>\LuaTeX</code>	LuaT _E X
<code>\LuaLaTeX</code>	LuaL ^A T _E X
<code>\XeTeX</code>	X _Ǝ T _E X, with reversed E if <code>graphics</code> is loaded.
<code>\XeLaTeX</code>	X _Ǝ L ^A T _E X, with reversed E if <code>graphics</code> is loaded.
<code>\AmS</code>	<i>AmS</i>
<code>\LyX</code>	LyX
<code>\BibTeX</code>	BibT _E X
<code>\MakeIndex</code>	<i>MakeIndex</i>
<code>\ConTeXt</code>	ConT _E Xt
<code>\MiKTeX</code>	MiKT _E X

3.16 Dashes and slashes

<code>\thinspace</code>	A breakable thin skip.
<code>\endash</code>	An endash: –
<code>\emdash</code>	An emdash: —
<code>\thinbrspace</code>	A thin space which allows a line break.
<code>\thinthinbrspace</code>	A very thin space which allows a line break.
<code>\Dash</code>	An unbreakable thin space, emdash, and breakable thin space: A—B
<code>\dash</code>	An unbreakable thin space, endash, and breakable thin space: A–B

\Slash

An unbreakable very thin space, a slash, and a breakable very thin space:

Command	Result	
A--B	A–B	(not breakable)
A \dash B	A–B	(only breakable before the B)
A -- B	A – B	(breakable before or after the dash)
A---B	A—B	(not breakable)
A \Dash B	A—B	(only breakable before the B)
A --- B	A — B	(breakable before or after the dash)
A/B	A/B	(not breakable)
A \Slash B	A/B	(only breakable before the B)
A / B	A / B	(breakable before or after the slash)
A~/~B	A / B	(not breakable)

4 Examples

Example 1: Macros

Code:

```
\DescribeMacro{\mymacro} \oarg{optional} \marg{mandatory}
  A typical macro definition.

\DescribeMacro[photograph]{\DeclareFloatingEnvironment}
Create a photograph float. \bigskip

\DescribeMacro[photograph]{\captionsetup}
Caption settings for a photograph float.

\DescribeMacro[photograph]{\cnameref}
\pkg{cleveref} name for the photograph float.
```

Result:

<code>\mymacro</code>	<code>[<i><optional></i>] [<i><mandatory></i>] A typical macro definition.</code>
<code>[photograph]</code> <code>\DeclareFloatingEnvironment</code>	Create a photograph float.
<code>[photograph] \captionsetup</code>	Caption settings for a photograph float.
<code>[photograph] \cnameref</code>	<code>cleveref</code> name for the photograph float.

The optional class is used to label and group tags and index entries. See this document's index entries for examples of this “photograph” class and the `dtxexample` class of macros.

[hyperlinks](#) The re-defined `\DescribeMacro`, `\DescribeEnv`, and all the following macros create hyperlinked index entries, along with regular uses of `\index`.

Example 2: Environment

Code:

`\DescribeEnv{myenvironment} \marg{argument} Short description.`

Result:

`{\langle argument \rangle} Short description.`

Env myenvironment

add'l tags
index groups

The re-defined `\DescribeEnv` adds an ‘Env’ tag to the margin, and adds “(environment)” to its own index entry. Note that environments and all the other new objects defined by this package each receives two index entries, one by name, and one grouped with others of its kind.

⚠ too much text

Example 2 shows descriptive text on the same line as the `\DescribeEnvironment`. For macros and environments with many arguments after the name, it may be better to place any additional text in a following paragraph.

Example 3: Second Environment

Code:

`\DescribeEnv[kindofenvironment]{otherenvironment}
 \oarg{opt args} \parg{coordinates} A description.`

Result:

`[\langle opt args \rangle] (\langle coordinates \rangle) A description.`

Env [kindofenvironment]
otherenvironment

The `otherenvironment` will be indexed by itself and also with `myenvironment` under the index entry “environments”, and also under the class `kindofenvironment`.

Example 4: Booleans and Counters

Code:

\DescribeBoolean[examples]{sampleboolean} Some description.

\DescribeCounter[examples]{samplecounter} Some description.

Result:

Bool [examples] sampleboolean

Some description.

Ctr [examples] samplecounter

Some description.

Most of the new \Describe_____ macros behave like the new \DescribeEnv, placing a tag in the margin, an index entry by name, and another index entry by group.

Example 5: Lengths

Code:

\DescribeLength[photograph]{\photowidth} Some description.

Result:

Len [photograph] \photowidth

Some description.

Lengths have a leading backslash, but are otherwise described the same as the rest of the objects.

Example 6: Packages, Classes, and Options

Code:

```
\DescribePackage[examples]{samplepackage}
  About a \LaTeX\ package.

\DescribeClass[examples]{sample_class}
  About a \LaTeX\ class.

\DescribeOption[examples]{sampleoption}
  About an option for a package or class.
```

Result:

Pkg [examples]	samplepackage	About a L ^A T _E X package.
Cls [examples]	sample_class	About a L ^A T _E X class.
Opt [examples]	sampleoption	About an option for a package or class.

Example 7: Files, Commands, and Programs

Code:

```
\DescribeFile[bigfiles]{really_big_file.txt} Some description.

\DescribeFile[bigfiles]{another_big_file.txt} Some description.

\DescribeFile{lone_file.txt} Some description.

\DescribeCommand{OS_command} An operating-system command.

\DescribeProgram{program_name} An operating-system program.
```

Result:

File	[bigfiles]	Some description.
really_big_file.txt		
File	[bigfiles]	Some description.
another_big_file.txt		
File		Some description.
lone_file.txt		
Cmd		An operating-system command.
OS_command		
Prog		An operating-system program.
program_name		

Filenames, program names, and command names may have underscores, such as tested here. A class is used to group “bigfiles” together in the index.

Example 8: Keys

Code:

```
\DescribeKey[groupofkeys]{firstkey} About the first key
of the |groupofkeys| set.

\DescribeKey[groupofkeys]{secondkey} About the second key
of |groupofkeys|.

\DescribeKey[examples]{samplekey} About some key of |otherkeys|.

\DescribeKey[examples]{sampletwokey} About another key of |otherkeys|.

\DescribeKey{lonekey} A key without a class.
```

Result:

Key [groupofkeys]	firstkey	About the first key of the groupofkeys set.
Key [groupofkeys]	secondkey	About the second key of groupofkeys.
Key [examples]	samplekey	About some key of otherkeys.
Key [examples]	sampletwokey	About another key of otherkeys.
Key	lonekey	A key without a class.

See the index key groups.

Example 9: Arguments

Code:

```
\DescribeArgument[figure]{[H]}
What happens when a figure is [H]ere.


\DescribeArgument[figure]{[M]}
What happens when a figure is in the [M]argin.

\DescribeArgument[\cs{mymacro}]{bold}
What happens when \cs{mymacro} is given the |bold| argument.
```

Result:

Arg [figure] [H]	What happens when a figure is [H]ere.
Arg [figure] [M]	What happens when a figure is in the [M]argin.
Arg [\mymacro] bold	What happens when \mymacro is given the bold argument.

Arguments behave like keys, and may have an optional class to identify their macro or environment, and group their entries in the index.

 **macro names** Note the need to use `\cs{mymacro}` for the macro's name.

Example 10: Object

Code:

```
\DescribeObject[color]{somecolor}
  The color of something.

\DescribeObject[color]{othercolor}
  The other color.

\DescribeObject{randomobject} About some random object.
```

Result:

```
[color] somecolor The color of something.
[color] othercolor The other color.
randomobject About some random object.
```

Describes an arbitrary programming object, using `\ttfamily` text.

Example 11: Other

Code:

```
\DescribeOther{license agreement}
The following is the fictional license agreement:

\DescribeOther{Before \env{myenvironment}}
  Actions to be done \cs{BeforeBeginEnvironment}.

\DescribeOther[otherclass]{Other Item} About the other item.

\DescribeOther[otherclass]{Additional Item} About the add'l item.
```

Result:

```
license agreement The following is the fictional license agreement:
Before myenvironment Actions to be done \BeforeBeginEnvironment.
[otherclass] Other Item About the other item.
[otherclass] Additional Item About the add'l item.
```

Describes an arbitrary non-programming object, using roman text.

Example 12: Description environments

Code:

```
\begin{description}
\ItemDescribeMacro[descexamples]{\macroname} Describe the macro.
\ItemDescribeBoolean[descexamples]{booleanname} Describe the boolean.
\ItemDescribeLength[descexamples]{\lengthname} Describe the length.
\ItemDescribeKey[descexamples]{keyname} Describe the key.
\ItemDescribePackage[descexamples]{package_name} Describe the package.
\ItemDescribeClass[descexamples]{class_name} Describe the class.
\ItemDescribeFile[descexamples]{file_name} Describe the file.
\ItemDescribeProgram[descexamples]{program_name} Describe the program.
\ItemDescribeCommand[descexamples]{command_name} Describe the class.
\end{description}
```

Result:

[descexamples] \macroname	\macroname: Describe the macro.
Bool [descexamples] booleanname	booleanname: Describe the boolean.
Len [descexamples] \lengthname	\lengthname: Describe the length.
Key [descexamples] keyname	keyname: Describe the key.
Pkg [descexamples] package_name	package_name: Describe the package.
Cls [descexamples] class_name	class_name: Describe the class.
File [descexamples] file_name	file_name: Describe the file.
Prog [descexamples] program_name	program_name: Describe the program.
Cmd [descexamples] command_name	command_name: Describe the class.

Uses a description environment to describe objects.

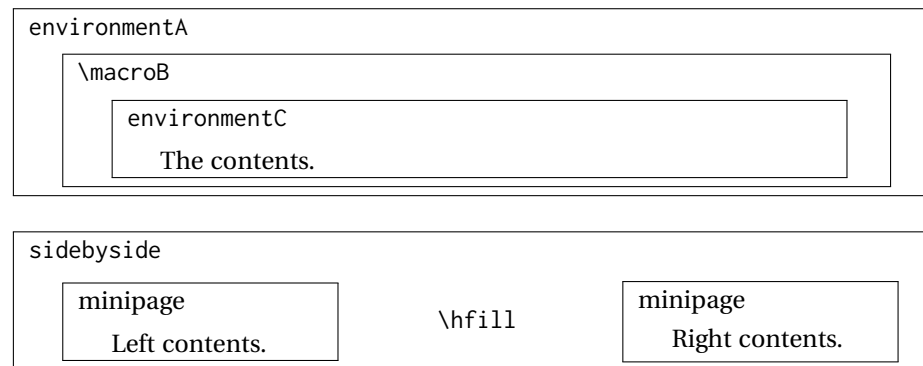
Example 13: Nesting

Code:

```
\shownesting{\env{environmentA}}{
  \shownesting{\cs{macroB}}{
    \shownesting{\env{environmentC}}{
      The contents.
    }
  }
}

\shownesting{\env{sidebyside}}{
  \shownesting[.35]{minipage}{
    Left contents.
  }
  \hfill \cs{hfill} \hfill
  \shownesting*[.35]{minipage}{
    Right contents.
  }
}
```

Result:



Note the use of the optional arguments to select less than full `\linewidth`, and the starred form for the second box to place it inline with the `\hfill` text.

Contents of the figure.

Figure 1: A Figure

Example 14: dtxexample*Code:*

```
\begin{figure}
  \centering\fbbox{Contents of the figure.}
  \caption{A Figure}\label{fig:afigure}
\end{figure}
```

*Result:**See fig. 1*

Example 14, typeset above, was created with the following code:

```
\begin{dtxexample}[See \cref{fig:afigure}]
  {\env{dtxexample}\label{ex:dtxexample}}
\begin{figure}
  \centering\fbbox{Contents of the figure.}
  \caption{A Figure}\label{fig:afigure}
\end{figure}
\end{dtxexample}
```

When the example was created:

1. The “float” of type example was created, with the caption “dtxexample” and the label `ex:dtxexample`, which points to example 14.
2. The code was displayed verbatim.
3. The code was written to the file `dtxexample_cut.tex`.
4. The code was `\input` from `dtxexample_cut.tex`.
5. Executing the code created the figure with caption “A Figure” and label `fig:afigure`, which points to fig. 1.
6. The cross-reference to the figure was shown on the optional display line by the optional argument to `dtxexample`.
7. The starred form of `dtxexample` was used to create the closing rule below the code, since a float was being generated and nothing followed the code inline. An unstarred version would have created an extra rule.

Example 15: fsourceverb*Code:*

```
% \begin{fsourceverb}[label=An fsourceverb example]
% \newcommand{fdosomething}[1][whattodo]{
%   doing #1
% }
% \end{fsourceverb}
```

Result:

```
_____ An fsourceverb example _____
\newcommand{fdosomething}[1][whattodo]{
  doing #1
}
```

(The leading % characters would be present in the dtx source.)

Example 16: sourcedisplay*Code:*

```
\begin{sourcedisplay}
\cs{newcommand}\{dosomething\}[1][\textcolor{red}{whattodo}]{\{\}
\quad \textcolor{blue}{doing \textcolor{red}{\#1}}}\}
\}
\end{sourcedisplay}
```

Result:

```
\newcommand{dosomething}[1][whattodo]{
  doing #1
}
```

Example 17: UIdisplay

Code:

```
Select:
\begin{UIdisplay}
  \textsf{Preferences $\to$ Plugins $\to$ Files $\to$ HTML}
\end{UIdisplay}
For the field
\begin{UIdisplay}
Title heading:
\end{UIdisplay}
\userentry{H1}
```

Result:

Select:

Preferences → Plugins → Files → HTML

For the field

Title heading:

Enter ⇒ **H1**

Example 18: docsidebar

Code:

Main text.

More main text.

```
\begin{docsidebar}[A title]
An aside, which may help explain something
incidental to the main text.
\end{docsidebar}
```

Additional main text.

Result:

Main text.

More main text.

A title

An aside, which may help explain something incidental to the main text.

Additional main text.

5 Usage notes

Placement of `\Describe` macros: Typically L^AT_EX macro and environment definitions are enclosed in macro and environment environments at their place in the source code. `\DescribeMacro` and `\DescribeEnv` would be used elsewhere in the manual to describe how to use the code. `\DescribeBoolean` and such might be at their place in the source code, unless they are worthy of discussion for the end-user, in which case they should be in the “User’s Manual” section of the document.¹ It may be useful to use `\DeclareBoolean` and friends both at the code location and also in the User’s Manual section.

Extra spaces: When placing multiple uses of `\Describe`, `\index`, `\margintag`, and `\watchout` macros together, care must be taken to avoid extra space in the printed text where these macros occur. A trailing percent character may be used to avoid the extra space:

```
text text text% <-- avoids extra space
\margintag{A comment.}
\index{An entry}
\index{Another entry}
more inline text
```

Unwanted vertical space: Other environments nested inside a `docsidebar` may produce excessive vertical space. It may be required to insert

```
\vspace*{-\baselineskip}
```

`\margintag` placement: To have the margin tag appear next to the first line of a paragraph, place the `\margintag` or `\watchout` somewhere after the first few words in the paragraph. The `\margintag` may be on its own line, and the rest of the paragraph may follow on the next line. If too many words are printed before the `\margintag`, the words may wrap to the next line before the tag occurs.

Margin tag overlap: To keep margin tags in proper alignment, use a new paragraph or multiple lines between `\margintag`, `\watchout`, or `\Declare` macros

missing tags **`\Describe` inside floats:** When these macros are used inside a float, the margin tag is suppressed (there is no margin in a float), but the index entries are still created.

¹Future versions may include `\DeclareBoolean` for use at the point where the boolean is defined, creating an index entry with a code line number, and `\DescribeBoolean` with a page number index entry for the related discussion in the User’s Manual portion of the document.

6 Code

6.1 Required packages

Pkg	makeidx	One of several index programs must be provided. One of several index programs must be provided.
Pkg	splitidx	
		<pre> 1 \AtBeginDocument{ 2 \ifpackageloaded{makeidx}{}{ 3 \ifpackageloaded{splitidx}{}{ 4 \RequirePackage{makeidx} 5 \makeindex 6 }} 7 }</pre>
Pkg	etoolbox	v2.6 or later for \BeforeBeginEnvironment, \AfterEndEnvironment
		<pre> 8 \RequirePackage{etoolbox}[2011/01/03]%</pre>
Pkg	xparse	Used for the examples.
		<pre> 9 \RequirePackage{xparse}</pre>
Pkg	calc	Used for \shownesting.
		<pre> 10 \RequirePackage{calc}</pre>
Pkg	xcolor	Used for the examples.
		<pre> 11 \RequirePackage{xcolor} 12 \definecolor{myurlcolor}{rgb}{0,0,.7} 13 \definecolor{mylinkcolor}{rgb}{.7,0,0}</pre>
Pkg	caption	Used for the examples.
		<pre> 14 \RequirePackage{caption}</pre>
Pkg	newfloat	Used for the examples.
		<pre> 15 \RequirePackage{newfloat}</pre>
Pkg	fancyvrb	Used for the examples.
		<pre> 16 \RequirePackage{fancyvrb}</pre>
Pkg	xstring	Used for \StrSubstitute for \DescribeFile.

Pkg hyperref

PDF bookmarks

If hyperref is loaded, disable some macros in PDF bookmarks:

```

17 \RequirePackage{xstring}

18 \AtBeginDocument{
19   \ifpackageloaded{hyperref}{
20     \pdfstringdefDisableCommands{%
21       \def\quad{ }%
22       \def\{ }%
23       \def\pkg#1{#1}%
24       \def\ctr#1{#1}%
25       \def\bool#1{#1}%
26       \def\optn#1{#1}%
27       \def\env#1{#1}%
28       \def\cs#1{\textbackslash#1}%
29       \def\,{ }%
30       \def\LuaLaTeX{LuaLaTeX}%
31       \def\XeLaTeX{XeLaTeX}%
32       \def\TeX{TeX}%
33       \def\LaTeX{LaTeX}%
34       \def\LaTeXe{LaTeX2e}%
35       \def\LuaTeX{LuaTeX}%
36       \def\LuaLaTeX{LuaLaTeX}%
37       \def\XeTeX{XeTeX}%
38       \def\AmS{AMS}%
39       \def\Dash{ --- }%
40       \def\dash{ -- }%
41       \def\Slash{/}%
42       \def\prog#1{\detokenize{#1}}%
43       \def\progcode#1{#1}%
44       \def\filenm#1{\detokenize{#1}}%
45       \def\brand#1{#1}%
46       \def\acro#1{#1}%
47       \def\ODT{ODT}%
48       \def\SVG{SVG}%
49       \def\PNG{PNG}%
50       \def\GIF{GIF}%
51       \def\JPG{JPG}%
52       \def\EPS{EPS}%
53       \def\PDF{PDF}%
54       \def\DVI{DVI}%
55       \def\UTF{UTF}%
56       \def\URL{URL}%
57       \def\element#1{#1}%
58       \def\attribute#1{#1}%
59       \def\attrib#1{#1}%
60       \def\HTML{HTML}%
61       \def\HTMLfive{HTML5}%
62       \def\CSS{CSS}%

```

```

63         \def\CSSthree{CSS3}%
64         \def\EPUB{EPUB}%
65         \def\TOC{TOC}%
66         \def\LOF{LOF}%
67         \def\LOT{LOT}%
68     }
69 }

```

If `hyperref` is not loaded, emulate `\hyperpage` here.

```

70 {
71     \newcommand*\hyperpage[1]{#1}
72 }
73 }

```

Pkg pict2e

```

74 \RequirePackage{pict2e}
75 \setlength{\unitlength}{1pt}

```

`\warningsign` Prints an exclamation point inside a triangle.

Creates a warning sign without relying on the presence of the fourier font. During copy/paste, this shows up as a simple exclamation point.

```

76 \newcommand*\warningsign{%
77 \begin{picture}(10,9)
78 \put(4,1){\scriptsize!}
79 \put(0,0){\line(500,866){5}}
80 \put(10,0){\line(-500,866){5}}
81 \put(0,0){\line(1,0){10}}
82 \end{picture}
83 }

```

6.2 Gobbling comment characters

`DTXD@gobble` The `.dtx` format uses leading percent characters for code to be in the documentation only. Other classes do not.

```

84 \ifpackageloaded{doc}{
85     \newcommand*\DTXD@gobble{2}
86 }{
87     \newcommand*\DTXD@gobble{0}
88 }

```

6.3 Vertical spacing

```

89 \setlength{\marginparsep}{1em}
90 \setlength{\marginparpush}{.7ex}
91
92 \setlength{\parindent}{0em}
93 \setlength{\parskip}{2ex}

```

From ltxdoc.

```

94 \ifdef{\IndexMin}
95   {\setlength{\IndexMin}{40ex}}
96   {\newlength{\IndexMin}}

```

6.4 ltxdoc emulation

If the ltxdoc class is not used, some of its macros are replicated here.

```

97 \ifclassloaded{ltxdoc}{}{
98   \def\cmd#1{\cs{\expandafter\cmd@to@cs\string#1}}
99   \def\cmd@to@cs#1#2{\char\number'#2\relax}
100   \DeclareRobustCommand\cs[1]{\texttt{\char'\#1}}
101   \providecommand\marg[1]{%
102     {\ttfamily\char'\{ }\meta{#1}{\ttfamily\char'\}}}
103   \providecommand\oarg[1]{%
104     {\ttfamily[ ]\meta{#1}{\ttfamily}}}
105   \providecommand\parg[1]{%
106     {\ttfamily( )\meta{#1}{\ttfamily}}}
107   \providecommand\url{\texttt}
108 }

```

6.5 doc emulation

If the doc class is not used, some of its macros are replicated here.

```

109 \AtBeginDocument{
110   \@ifpackageloaded{doc}{}{
111     \newenvironment*{macro}[1]{%
112       \PackageError{dtxdescribe}
113         {The 'macro' environment is only\MessageBreak
114           available when using the doc package\MessageBreak
115           with a .dtx source file}
116       {This environment only makes sense for .dtx source.}
117     }{}
118     \newenvironment*{environment}[1]{%
119       \PackageError{dtxdescribe}
120       {The 'environment' environment is only\MessageBreak

```

Len \IndexMin

```

121         available when using the doc package\MessageBreak
122         with a .dtx source file}
123         {This environment only makes sense for .dtx source.}
124     {}
125     \def\MacroFont{\fontencoding\encodingdefault
126         \fontfamily\ttdefault
127         \fontseries\mddefault
128         \fontshape\updefault
129         \small}%
130     \@ifundefined{actualchar}{\def\actualchar{@}}{}
131     \@ifundefined{quotechar}{\def\quotechar{"}}{}
132     \@ifundefined{levelchar}{\def\levelchar{!}}{}
133     \@ifundefined{encapchar}{\def\encapchar{[]}}{}
134     \@ifundefined{verbatimchar}{\def\verbatimchar{+}}{}
135     \setlength\marginparpush{0pt} \setlength\marginparwidth{8pc}
136     \reversemarginpar
137     \DeclareRobustCommand\meta[1]{%
138         \ensuremath\langle
139         \ifmmode \expandafter \nfss@text \fi
140         {%
141             \meta@font@select
142             \edef\meta@hyphen@restore
143                 {\hyphenchar\the\font\the\hyphenchar\font}%
144             \hyphenchar\font\m@ne
145             \language\l@nohyphenation
146             #1\/%
147             \meta@hyphen@restore
148         }\ensuremath\rangle
149     }
150     \def\meta@font@select{\itshape}
151 }
152 }

```

6.6 Support macros

`\PrintEnvName` $\{\langle name \rangle\}$ Prints an environment name.

```

153 \providecommand*\PrintEnvName{}
154 \renewcommand*\PrintEnvName[1]
155 {\strut{\scriptsize{}Env}\quad\MacroFont#1\ }

```

`\DTXD@printtype` $\{\langle text \rangle\}$

Used to print the object class in the margin:

```

156 \newcommand*\DTXD@printtype[1]
157 {\raggedleft\strut{\scriptsize\sffamily#1}\quad\MacroFont}

```

`\usage` $\{\langle text \rangle\}$

Allow hyperlinks in the “usage” index entries:

```
158 \providecommand{\usage}{}
159 \renewcommand{\usage}[1]{\textit{\hyperpage{#1}}}
```

`\DTXD@origwindex` Used to bypass `hyperref` index modifications.

```
160 \let\DTXD@origwindex\@windex
```

`\DTXD@margin` $\tag{ \{\langle class \rangle\} \{\langle name \rangle\} \{\langle margin tag \rangle\}}$

Creates the margin tag for the object being described.

The `class` is used to sub-categories keys into their key/value groups.

```
161 \newcommand*{\DTXD@margin}[3]{%
162   \@ifundefined{@capttype}{% not float?
163     \leavevmode%
164     \marginpar{%
165       {%
166         \hbadness=10000%
167         \hfuzz=5em%
168         \DTXD@printtype{%
169           #3% margin tag
170           \ifblank{#1}{\{ [#1]\}% class
171         }%
172         \texttt{#2}% name
173       }%
174     }% marginpar
175   }\}% not float?
176 }
```

`\DTXD@index` $\{\langle class \rangle\} \{\langle name \rangle\} \{\langle margin tag \rangle\} \{\langle index tag \rangle\} \{\langle main/usage \rangle\}$

Creates the index entries for the object being described, where `name` has no backslash or underscore.

The `class` is used to sub-categories keys into their key/value groups. `main` prints code lines in the index, and `usage` prints page numbers.

```
177 \newcommand*{\DTXD@index}[5]{%
```

The `makeindex` program allows each index entry to call a macro by appending a vertical bar and a macro name to each entry. `hyperref` adds a call by `\hyperpage` to each index entry, by appending the phrase `|hyperpage` to the entry in the `.idx` file.

The `doc` package uses the same mechanism to distinguish between code line entries (`|main`) and references to the use of a macro (`|usage`). The problem is that `makeindex` can only handle one macro call, but `hyperref` tries to append its `|hyperpage` to the already-existing `|usage` or `|main`.

The solution used for `dtxdescribe` is to allow `hyperref` to modify all regular index entries, but use the original definition of `\@wrindex` for the `\Describe_____` macros, before `hyperref` modified it. Then, the `\usage` macro, defined above, manually adds the hyperlink.

Below, `\@bsphack` and `\@esphack` seem to be required for `\@wrindex` to work. `\ignorespaces` is used in addition because `\Declare` and `\index` entries often come in groups.

```
178 \@bsphack%
179 \begingroup%
180 \DTXD@origwrindex{%
```

Index by name:

Write the name, the formatted name, the index tag, and the class:

```
181 #2\actualchar{\protect\ttfamily#2} % name
182 (#4)% index tag
183 \ifblank{#1}{}{ [#1]}% class
184 \encapchar #5}%
```

Index by tag and class:

Write the tag and class as a group, under which is the name and the formatted name.

```
185 \begingroup%
186 \DTXD@origwrindex{%
187 #4:\levelchar% index tag
188 \ifblank{#1}{}{[#1]:\levelchar}% class
189 #2\actualchar{\protect\ttfamily#2}% name
190 \encapchar #5}%
```

Possibly index by class and name:

```
191 \ifblank{#1}{}{% class given
192 \begingroup%
193 \DTXD@origwrindex{%
194 #1\actualchar[#1]:\levelchar% class
195 #2\actualchar{\protect\ttfamily#2} % name
196 (#4)% index tag
197 \encapchar #5}%
```

```

198 }% class given
199 % \@esphack%
200 \@esphack%
201 \ignorespaces%
202 }

```

`\DTXD@marginindex` $\{\langle class \rangle\} \{\langle name \rangle\} \{\langle margin\ tag \rangle\} \{\langle index\ tag \rangle\} \{\langle main/usage \rangle\}$

Creates the margin tag and the index entries. The class is used to sub-categories keys into their key/value groups.

```

203 \newcommand*{\DTXD@marginindex}[5]{%
204 % \@bsphack%

```

The margin tag and the name:

```

205 \DTXD@margin{#1}{#2}{#3}%

```

The index entries:

```

206 \DTXD@index{#1}{#2}{#3}{#4}{#5}%
207 }

```

`\DTXD@macroname` $\{\langle control\ sequence \rangle\}$

Given a control sequence such as `\name`, prints its name without the backslash.

From: <http://tex.stackexchange.com/questions/42318/removing-a-backslash-from-a-character-sequence>

```

208 \begingroup\lccode'\|='\\
209 \lowercase{\endgroup\def\removebs#1{\if#1|\else#1\fi}}
210 \newcommand*{\DTXD@macroname}[1]{\expandafter\removebs\string#1}

```

`\DTXD@verbatimcmd` $\{\langle \backslash name \rangle\}$

While printing to the index file, prints the `\name` verbatim. From `\SpecialIndex` in the `doc` package.

```

211 \newcommand*{\DTXD@verbatimcmd}[1]{%
212 \string\verb\quotechar*\verbatimchar\string#1\verbatimchar%
213 }

```

`\DTXD@cmdmarginindex` $\{\langle class \rangle\} \{\langle name \rangle\} \{\langle margin\ tag \rangle\} \{\langle index\ tag \rangle\} \{\langle main/usage \rangle\}$

Creates the margin tag and index entries where name is a `\macro`.


```

214 \newcommand*{\DTXD@cmdmarginindex}[5]{%
215 \@bsphack%

```

Create a margin tag with the name of the macro:

```

216 \@ifundefined{@capttype}{% not float?
217 \leavevmode%
218 \marginpar{%
219   {%
220     \hbadness=10000%
221     \hfuzz=5em%
222     \DTXD@printtype{%
223       #3% margin tag
224       \ifblank{#1}{\{ [#1]}% class
225     }%
226     \cmd{#2}% name
227   }%
228 }% marginpar
229 }{\}% not float?

```

Create an index entry sorted by the name without its leading backslash, followed by the macro name with the backslash, and the tag. Prepend with the class if given.

Write (class):>name=csname (indextag)|usage

```

230 \begingroup%
231 \DTXD@origwindex{%
232 \ifblank{#1}{\{#1\actualchar[#1]:\levelchar}% class
233 \DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2} % name
234 (#4)% index tag
235 \encapchar #5}%

```

Create an index entry grouped by the tag, then printed and sorted by the macro name with the backslash, and the tag.

Write indextag:>(class):>csname|usage

```

236 \begingroup%
237 \DTXD@origwindex{%
238 #4:\levelchar% index tag
239 \ifblank{#1}{\{[#1]:\levelchar}% class
240 \DTXD@verbatimcmd{#2}% name
241 \encapchar #5}%
242 \@esphack%
243 \ignorespaces%
244 }

```

6.7 \DescribeMacro and \DescribeEnvironment

\DescribeMacro [*<class>*] {<\name>}

Redefined to allow hyperlinked index entries and an optional class:

```
245 \providecommand*\DescribeMacro{}{}
246 \renewcommand*\DescribeMacro[2][{}]{%
247 \@bsphack%
```

Create the margin tag with the macro's name:

```
248 \@ifundefined{@capttype}{% not float?
249 \leavevmode%
250 \marginpar{%
251   {%
252     \hbadness=10000%
253     \hfuzz=5em%
254     \raggedleft%
255     \ifblank{#1}{}{{\scriptsize\textsf{[#1]}} }% class
256     \cmd{#2}% name
257   }%
258 }% marginpar
259 }{}% not float?
```

Write the index sorted by the name without the backslash, followed by the actual name with the backslash. Append the class if given.

Write name=csname>(class)|usage

```
260 \begingroup%
261 \DTXD@origwindex{%
262   \DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2}% name
263   \ifblank{#1}{}{\levelchar[#1]}% class
264   \encapchar usage%
265 }%
```

Only if a class was given:

```
266 \ifblank{#1}%
267 {}% no class
268 {% class given
269 % Again, and prepend the class:
270 %
271 % Write class=(class):>name=csname\verb+|usage+
272 %   \begin{macrocode}
273   \begingroup%
274   \DTXD@origwindex{%
```

```

275    #1\actualchar[#1]:\levelchar%
276    \DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2}%
277    \encapchar usage}%
278 }% class given
279 \@esphack%
280 \ignorespaces%
281 }

```

`\DescribeEnv` [*⟨class⟩*] {*⟨environment name⟩*}

Redefined to allow hyperlinked index entries:

```

282 \providecommand*\DescribeEnv{}
283 \renewcommand*\DescribeEnv[2][]
284 {\DTXD@margintagindex{#1}{#2}{Env}{environment}{usage}}

```

6.8 New `\Describe. . . macros`

`\DTX@filename` Stores the filename with a sanitized underscore.

```

285 \newcommand*\DTXD@filename{}

```

`\DTXD@filemarginparindex` {*⟨class⟩*} {*⟨name⟩*} {*⟨margin tag⟩*} {*⟨index tag⟩*} {*⟨main/usage⟩*}

The name may have underscores.

```

286 \newcommand*\DTXD@filemarginparindex[5]{%

```

Create a detokenized version of the filename...

```

287 \renewcommand*\DTXD@filename{\detokenize{#2}}%

```

... then replace any underscores with a detokenized `_`, which will print as an underscore when read back from the index file:

```

288 \StrSubstitute{\DTXD@filename}%
289 {\detokenize{\_}}{\detokenize{\_}}[\DTXD@filename]%

```

The original filename is printed in the margin. Any underscore characters have already been disabled by the `\catcode` change.

```

290 \DTXD@margintag{#1}{#2}{#3}%

```

The detokenized and sanitized version is sent to the index file:

```

291 \DTXD@index{#1}{\DTXD@filename}{#3}{#4}{#5}%

```

End the group with the disabled underscore, and clean up the extra space from the `\catcode` command:

```
292 \endgroup%
293 \ignorespaces%
294 }
```

`\DTXD@DescribeFile` [`\class`] {`\name`}

The name may have underscores.

```
295 \newcommand*{\DTXD@DescribeFile}[2][{}]{%
296 \DTXD@filemarginparindex{#1}{#2}{File}{file}{usage}%
297 }
```

`\DescribeFile` {`\name`}

The underscore character is temporarily disabled, then the name is passed directly to `\DTXD@DescribeFile`.

```
298 \newcommand*{\DescribeFile}{%
299 \begingroup\catcode'\_ =12 \DTXD@DescribeFile%
300 }
```

`\DTXD@DescribeProgram` [`\class`] {`\name`}

The name may have underscores.

```
301 \newcommand*{\DTXD@DescribeProgram}[2][{}]{%
302 \DTXD@filemarginparindex{#1}{#2}{Prog}{program}{usage}%
303 }
```

`\DescribeProgram` {`\name`}

The underscore character is temporarily disabled, then the name is passed directly to `\DTXD@DescribeProgram`.

```
304 \newcommand*{\DescribeProgram}{%
305 \begingroup\catcode'\_ =12 \DTXD@DescribeProgram%
306 }
```

`\DTXD@DescribeCommand` [`\class`] {`\name`}

The name may have underscores.

```
307 \newcommand*{\DTXD@DescribeCommand}[2][{}]{%
```

```
308 \DTXD@filemarginparindex{#1}{#2}{Cmd}{command}{usage}%
309 }
```

`\DescribeCommand` { *$\langle name \rangle$* }

The underscore character is temporarily disabled, then the name is passed directly to `\DTXD@DescribeCommand`.

```
310 \newcommand*{\DescribeCommand}{%
311 \begingroup\catcode'\_ =12 \DTXD@DescribeCommand%
312 }
```

`\DTXD@DescribePackage` [*$\langle class \rangle$*] { *$\langle name \rangle$* } The name may have underscores.

```
313 \newcommand*{\DTXD@DescribePackage}[2][{}]{%
314 \DTXD@filemarginparindex{#1}{#2}{Pkg}{package}{usage}%
315 }
```

`\DescribePackage` { *$\langle name \rangle$* }

The underscore character is temporarily disabled, then the name is passed directly to `\DTXD@DescribePackage`.

```
316 \newcommand*{\DescribePackage}{%
317 \begingroup\catcode'\_ =12 \DTXD@DescribePackage%
318 }
```

`\DTXD@DescribeClass` [*$\langle class \rangle$*] { *$\langle name \rangle$* }

The name may have underscores.

```
319 \newcommand*{\DTXD@DescribeClass}[2][{}]{%
320 \DTXD@filemarginparindex{#1}{#2}{Cls}{class}{usage}%
321 }
```

`\DescribeClass` { *$\langle name \rangle$* }

The underscore character is temporarily disabled, then the name is passed directly to `\DTXD@DescribeClass`.

```
322 \newcommand*{\DescribeClass}{%
323 \begingroup\catcode'\_ =12 \DTXD@DescribeClass%
324 }
```

`\DescribeOption` [*$\langle class \rangle$*] { *$\langle name \rangle$* }

```
325 \newcommand*{\DescribeOption}[2][]
326 {\DTXD@margintagindex{#1}{#2}{Opt}{option}{usage}}
```

`\DescribeArgument` [*⟨class⟩*] {*⟨name⟩*}

The class may be used to categorize arguments by their macro or environment name.

```
327 \newcommand*{\DescribeArgument}[2][]
328 {\DTXD@margintagindex{#1}{#2}{Arg}{argument}{usage}}
```

`\DescribeBoolean` [*⟨class⟩*] {*⟨name⟩*}

```
329 \newcommand*{\DescribeBoolean}[2][]
330 {\DTXD@margintagindex{#1}{#2}{Bool}{boolean}{usage}}
```

`\DescribeLength` [*⟨class⟩*] {*⟨name⟩*}

```
331 \newcommand*{\DescribeLength}[2][]
332 {\DTXD@cmdmargintagindex{#1}{#2}{Len}{length}{usage}}
```

`\DescribeCounter` [*⟨class⟩*] {*⟨name⟩*}

```
333 \newcommand*{\DescribeCounter}[2][]
334 {\DTXD@margintagindex{#1}{#2}{Ctr}{counter}{usage}}
```

`\DescribeHook` [*⟨class⟩*] {*⟨name⟩*}

```
335 \newcommand*{\DescribeHook}[2][]
336 {\DTXD@margintagindex{#1}{#2}{Hook}{hook}{usage}}
```

`\DescribeKey` [*⟨class⟩*] {*⟨name⟩*}

The class may be used to categorize keys by their key/value group.

```
337 \newcommand*{\DescribeKey}[2][]
338 {\DTXD@margintagindex{#1}{#2}{Key}{key}{usage}}
```

`\DescribeObject` [*⟨class⟩*] {*⟨name⟩*}

May be used to describe an arbitrary piece of code. Creates a margin tag and index entries with `\ttfamily`.

```
339 \newcommand*{\DescribeObject}[2][\%
```

```

340 \ifundefined{@capttype}{% not float?
341     \@bsphack%
342     \leavevmode%
343     \marginpar{%
344         \hbadness=10000%
345         \hfuzz=5em%
346         \raggedleft%
347         \ifblank{#1}{\raggedleft\scriptsize[#1]} }
348         \texttt{#2}%
349     }%
350 }{% not float?
351 \ifblank{#1}%
352 {%
353     \begingroup%
354     \DTXD@origwindex{%
355         #2\actualchar{\protect\ttfamily#2}%
356         \encapchar usage%
357     }%
358 }%
359 {%
360     \begingroup%
361     \DTXD@origwindex{%
362         #2\actualchar{\protect\ttfamily#2} [#1]%
363         \encapchar usage%
364     }%
365     \begingroup%
366     \DTXD@origwindex{%
367         #1\actualchar[#1]:\levelchar#2\actualchar{\protect\ttfamily#2}%
368         \encapchar usage%
369     }%
370 }%
371 \@esphack%
372 \ignorespaces%
373 }

```

`\DescribeOther` [*⟨class⟩*] {*⟨name⟩*}

May be used to describe an arbitrary non-programming object. Creates a margin tag and index entries with roman type.

```

374 \newcommand*{\DescribeOther}[2][]{%
375 \ifundefined{@capttype}{% not float?
376     \@bsphack%
377     \leavevmode%
378     \marginpar{%
379         \hbadness=10000%
380         \hfuzz=5em%
381         \raggedleft%
382         \ifblank{#1}{\raggedleft\scriptsize[#1]} }%

```

```

383      #2%
384    }%
385 }{}% not float?
386 \ifblank{#1}%
387 {%
388   \begingroup%
389   \DTXD@origwindex{#2\encapchar usage}%
390 }%
391 {%
392   \begingroup%
393   \DTXD@origwindex{#2 [#1]\encapchar usage}%
394   \begingroup%
395   \DTXD@origwindex{#1\actualchar[#1]:\levelchar#2\encapchar usage}%
396 }%
397 \@esphack%
398 \ignorespaces%
399 }

```

6.9 \DescribeDefault

`\DescribeDefaultcolor` The color of the margin tag used to show the default value.

```
400 \newcommand*{\DescribeDefaultcolor}{green!50!black}
```

`\DescribeDefault` $\{\langle value \rangle\}$

Creates a colored margin tag showing the booleandefault value.

```

401 \newcommand{\DescribeDefault}[1]{%
402   \margintag{%
403     \footnotesize%
404     \textcolor{\DescribeDefaultcolor}{%
405       Default: \texttt{#1}%
406     }%
407   }%
408 }

```

6.10 \ItemDescribeMacro, etc.

The following are for use inside a description.

`\ItemDescribeMacro` $[\langle class \rangle] \{\langle name \rangle\}$

```
409 \newcommand{\ItemDescribeMacro}[2][{}]{%
```



```

410 \item[\cmd{#2}:]%
411 \setlength{\parskip}{1.5ex}%
412 \DescribeMacro[#1]{#2}%
413 }

```

`\ItemDescribeEnv` [*⟨class⟩*] {*⟨name⟩*}

```

414 \newcommand{\ItemDescribeEnv}[2][]{%
415 \item[\env{#2}:]%
416 \setlength{\parskip}{1.5ex}%
417 \DescribeEnv[#1]{#2}%
418 }

```

`\ItemDescribeArgument` [*⟨class⟩*] {*⟨argument⟩*}

```

419 \newcommand{\ItemDescribeArgument}[2][]{%
420 \item[\texttt{#2}:]%
421 \setlength{\parskip}{1.5ex}%
422 \DescribeArgument[#1]{#2}%
423 }

```

`\ItemDescribeBoolean` [*⟨class⟩*] {*⟨name⟩*}

```

424 \newcommand{\ItemDescribeBoolean}[2][]{%
425 \item[\texttt{#2}:]%
426 \setlength{\parskip}{1.5ex}%
427 \DescribeBoolean[#1]{#2}%
428 }

```

`\ItemDescribeLength` [*⟨class⟩*] {*⟨name⟩*}

```

429 \newcommand{\ItemDescribeLength}[2][]{%
430 \item[\cmd{#2}:]%
431 \setlength{\parskip}{1.5ex}%
432 \DescribeLength[#1]{#2}%
433 }

```

`\ItemDescribeCounter` [*⟨class⟩*] {*⟨name⟩*}

```

434 \newcommand{\ItemDescribeCounter}[2][]{%
435 \item[\texttt{#2}:]%
436 \setlength{\parskip}{1.5ex}%
437 \DescribeCounter[#1]{#2}%
438 }

```

`\ItemDescribeHook` [`<class>`] {`<name>`}

```
439 \newcommand{\ItemDescribeHook}[2][{}]{%
440   \item[\texttt{#2}:]%
441   \setlength{\parskip}{1.5ex}%
442   \DescribeHook[#1]{#2}%
443 }
```

`\ItemDescribeKey` [`<class>`] {`<name>`}

```
444 \newcommand{\ItemDescribeKey}[2][{}]{%
445   \item[\texttt{#2}:]%
446   \setlength{\parskip}{1.5ex}%
447   \DescribeKey[#1]{#2}%
448 }
```

`\ItemDescribePackage` [`<class>`] {`<name>`}

```
449 \newcommand{\DTXD@ItemDescribePackage}[2][{}]{%
450   \item[\texttt{#2}:]%
451   \setlength{\parskip}{1.5ex}%
452   \DescribePackage[#1]{#2}%
453   \endgroup%
454 }
455
456 \newcommand{\ItemDescribePackage}{%
457   \begingroup\catcode'\_ =12 \DTXD@ItemDescribePackage%
458 }
```

`\ItemDescribeClass` [`<class>`] {`<name>`}

```
459 \newcommand{\DTXD@ItemDescribeClass}[2][{}]{%
460   \item[\texttt{#2}:]%
461   \setlength{\parskip}{1.5ex}%
462   \DescribeClass[#1]{#2}%
463   \endgroup%
464 }
465
466 \newcommand{\ItemDescribeClass}{%
467   \begingroup\catcode'\_ =12 \DTXD@ItemDescribeClass%
468 }
```

`\ItemDescribeOption` [`<class>`] {`<name>`}

```
469 \newcommand{\ItemDescribeOption}[2][{}]{%
470   \item[\texttt{#2}:]%
471   \setlength{\parskip}{1.5ex}%
472 }
```

```

472   \DescribeOption[#1]{#2}%
473 }

```

`\ItemDescribeFile` [*⟨class⟩*] {*⟨name⟩*}

```

474 \newcommand{\DTXD@ItemDescribeFile}[2][{}]{%
475   \item[\texttt{#2}:]%
476   \setlength{\parskip}{1.5ex}%
477   \DescribeFile[#1]{#2}%
478   \endgroup%
479 }
480
481 \newcommand{\ItemDescribeFile}{%
482   \begingroup\catcode'\_ =12 \DTXD@ItemDescribeFile%
483 }

```

`\ItemDescribeProgram` [*⟨class⟩*] {*⟨name⟩*}

```

484 \newcommand{\DTXD@ItemDescribeProgram}[2][{}]{%
485   \item[\texttt{#2}:]%
486   \setlength{\parskip}{1.5ex}%
487   \DescribeProgram[#1]{#2}%
488   \endgroup%
489 }
490
491 \newcommand{\ItemDescribeProgram}{%
492   \begingroup\catcode'\_ =12 \DTXD@ItemDescribeProgram%
493 }

```

`\ItemDescribeCommand` [*⟨class⟩*] {*⟨name⟩*}

```

494 \newcommand{\DTXD@ItemDescribeCommand}[2][{}]{%
495   \item[\texttt{#2}:]%
496   \setlength{\parskip}{1.5ex}%
497   \DescribeCommand[#1]{#2}%
498   \endgroup%
499 }
500
501 \newcommand{\ItemDescribeCommand}{%
502   \begingroup\catcode'\_ =12 \DTXD@ItemDescribeCommand%
503 }

```

`\ItemDescribeObject` [*⟨class⟩*] {*⟨name⟩*}

```

504 \newcommand{\ItemDescribeObject}[2][{}]{%
505   \item[\texttt{#2}:]%
506   \setlength{\parskip}{1.5ex}%

```

```

507   \DescribeObject[#1]{#2}%
508 }

```

`\ItemDescribeOther` [`<class>`] {`<name>`}

```

509 \newcommand{\ItemDescribeOther}[2][]{%
510   \item[\texttt{#2}:]%
511   \setlength{\parskip}{1.5ex}%
512   \DescribeOther[#1]{#2}%
513 }

```

6.11 `\margintag`, `\watchout`

`\margintagcolor` The color of the `\margintag`.

```

514 \newcommand*{\margintagcolor}{blue!70!black}

```

`\margintag` {`<text>`}

Prints a colored margin tag.

```

515 \newcommand{\margintag}[1]{%
516 \@ifundefined{@capttype}{% not float?
517 \marginpar{\raggedleft\textcolor{\margintagcolor}{#1}}%
518 \ignorespaces%
519 }{}% not float?
520 }

```

`\watchoutcolor` The color of the `\watchout`.

```

521 \newcommand*{\watchoutcolor}{red!50!black}

```

`\watchout` [`<text>`]

Prints a warning sign and optional text.

```

522 \newcommand{\watchout}[1][]{%
523 \@ifundefined{@capttype}{% not float?
524   \marginpar{%
525     \raggedleft%
526     \textcolor{\watchoutcolor}{\warningsign\normalsize\quad#1}%
527   }%
528   \ignorespaces%
529 }{}% not float?
530 }

```

6.12 Nesting

Shows a box enclosing a label for the container, and the container's contents. May be nested.

`\shownesting` [*$\langle fraction of \linewidth \rangle$*] [*$\langle container \rangle$*] [*$\langle contents \rangle$*]

```

531 \NewDocumentCommand{\shownesting}{s O{1} m m}{
532   \IfBooleanF{#1}{
533     \par\smallskip
534   }
535   \fbox{
536     \begin{minipage}{#2\linewidth-2em}
537       \hbadness=10000\relax%
538       #3\par\smallskip
539       \hspace{1em}
540       \begin{minipage}{\linewidth-1.5em}
541         #4
542       \end{minipage}
543     \end{minipage}
544   }
545 }
```

6.13 The dtxexample environment

Also see example 14 on page page 28.

File `dtxexample_cut.tex` Used to store then `\input` example code.

[color] The color of the middle rule in the dtxexample.
DTXD@examplerulecolor
546 \definecolor{DTXD@examplerulecolor}{rgb}{.9,.9,.9}

`\dtxexamplecodename` The text name of the code section.

```

547 \newcommand*{\dtxexamplecodename}{Code:}
```

`\dtxexampleresultname` The text name of the result section.

```

548 \newcommand*{\dtxexampleresultname}{Result:}
```

Env `dtxexample` * [*$\langle notes/cross-references \rangle$*] [*$\langle caption \& label \rangle$*]

Reads the code listing as a verbatim input using the `fancybox` package, then displays the code listing as a verbatim output, and also executes the code and displays the

result. A title caption is specified, along with optional cross-referencing commands or notes to refer to the results. The unstarred version places the code inside a minipage, forbidding a page break in the middle of the code listing. The starred version does not use a minipage. This is required when the code is too large to fit on a single page.

```
549 \NewDocumentEnvironment{dtxexample}{s +0{} m}
550 {% start dtxexample
```

Copy the environment's contents to the file `dtxexample_cut.tex`:

```
551 \VerbatimOut[gobble=\DTXD@gobble,tabsize=4]{dtxexample_cut.tex}%
552 }% start dtxexample
```

When the environment closes:

```
553 {% end dtxexample
```

Finish the verbatim output:

```
554 \endVerbatimOut
555 \par
556 \addvspace{\bigskipamount}
```

If unstarred, typeset the example in a minipage:

```
557 \IfBooleanTF{#1}{\vspace{\bigskipamount}}{\minipage{\linewidth}}%
```

Emulated a float of type “example”:

```
558 \captionsetup{type=dtxdexample}%
559 \hrule\medskip
560 \caption{#3}
```

Typeset the contents as verbatim:

```
561 \textcolor{DTXD@examplerulecolor}{\smallskip\hrule}
562 \smallskip
563 {\scriptsize\itshape\dtxexamplecodename}
564 \VerbatimInput[tabsize=4]{dtxexample_cut.tex}
565 \unskip
566 \textcolor{DTXD@examplerulecolor}{\hrule}
567 \smallskip
568 {\scriptsize\itshape\dtxexampleresultname}
569
```

Possible add the optional cross-references or notes:

```

570 \ifstrempy{#2}
571 {}
572 {\itshape\small #2}}

```

If unstarred, close the `\minipage`.

```

573 \IfBooleanTF{#1}{}\endminipage}%
574 } % end dtxexample

```

Outside of the environment's scope, input the example to generate its output and labels:

```

575 \AfterEndEnvironment{dtxexample}
576 {%

```

Execute the code:

```

577 \par\unskip\input{dtxexample_cut.tex}%

```

Closing rule::

```

578 \medskip\hrule%
579 }

```

[dtxexample] A new float type for the examples.
`\DeclareFloatingEnvironment`

```

580 \DeclareFloatingEnvironment[
581 fileext=lox,
582 listname={List of Examples},
583 name=Example,
584 placement=hbpb
585 ]{dtxdexample}

```

[dtxexample] `\captionsetup` Caption setup for the examples.

```

586 \captionsetup*[dtxdexample]{
587 format=hang,
588 font=bf,
589 justification=raggedright,
590 singlelinecheck=false,
591 skip=0pt,
592 position=top,
593 }

```

[dtxexample] `\crefname` Name for `cleveref`.

```

594 \AtBeginDocument{

```

```

595 \@ifpackageloaded{cleveref}{\crefname{dtxdexample}{example}{examples}}{}
596 }

```

6.14 noindmacro and noindenvironment

Similar to macro and environment, but not indexed.

Env noindmacro $\{\langle name \rangle\}$

```

597 \newenvironment{noindmacro}[1]
598 {
599     \setlength{\parskip}{\marginparpush}
600     \leavevmode\par\DTXD@margintag{}\cmd{#1}}{}
601 }
602 {\unskip}

```

Env noindenvironment $\{\langle name \rangle\}$

```

603 \newenvironment{noindenvironment}[1]
604 {
605     \setlength{\parskip}{\marginparpush}
606     \leavevmode\par\DTXD@margintag{}\cmd{#1}{Env}
607 }
608 {\unskip}

```

6.15 sourcedisplay, UIdisplay, docsidebar

For use in a sourcedisplay:

\backslash fquad Forces a quad indent.

```

609 \newcommand*\fquad{\hspace*{1em}}

```

\backslash fqqquad Forces a double-quad indent.

```

610 \newcommand*\fqqquad{\hspace*{2em}}

```

\backslash fqqquad Forces a triple-quad indent.

```

611 \newcommand*\fqqquad{\hspace*{3em}}

```


Env sourceverb To typeset a block of source code, verbatim.

```
612 \DefineVerbatimEnvironment{sourceverb}{Verbatim}
613     {gobble=\DTXD@gobble,tabsize=4,xleftmargin=2em}
614 \BeforeBeginEnvironment{sourceverb}{\vspace*{-.5\parskip}}
```

Env fsourceverb To typeset a framed block of source code, verbatim.

```
615     \DefineVerbatimEnvironment{fsourceverb}{Verbatim}
616         {gobble=\DTXD@gobble,tabsize=4,xleftmargin=2em,frame=lines}
617 \BeforeBeginEnvironment{fsourceverb}{\vspace*{-.5\parskip}}
```

Env sourcedisplay To typeset a block of source code, allowing direct formatting.

```
618 \newenvironment{sourcedisplay}
619 {
620     \leavevmode
621     \par
622     \fqquad\minipage{\linewidth-4em}
623     \ttfamily
624 }
625 {%
626     \endminipage
627     \par
628 }
```

Env UIdisplay To typeset a user interface display.

```
629 \newenvironment{UIdisplay}
630 {
631     \leavevmode
632     \par
633     \fqquad\minipage{\linewidth-4em}
634     \sffamily\bfseries
635 }
636 {
637     \endminipage
638     \par
639 }
```

\userentryname Text to tell the user to enter the following item.

```
640 \newcommand*{\userentryname}{Enter~$\rightarrow$}
```

\userentry {*{text to enter}*}

Typesets text to be entered by the users.

```

641 \newcommand{\userentry}[1]{%
642 \par
643 \fqquad%
644 \begin{minipage}{\linewidth-2em}
645   {\footnotesize \userentryname}\quad\cmds{#1}
646 \end{minipage}
647 \par
648 }
```

Env docsidebar To typeset a sidebar in the documentation.

```

649 \newenvironment{docsidebar}[1][ ]
650 {%
651   \quote\unskip\medskip
652   \setlength{\parskip}{1.5ex}%
653   \ifblank{#1}{}{\textit{#1}\newline}%
654   \rule[.5\bigskipamount]{\linewidth}{.4pt}%
655   \newline%
656 }
657 {%
658   \leavevmode\par
659   \rule[\bigskipamount]{\linewidth}{.4pt}
660   \endquote\unskip
661 }
```

6.16 Formatted objects

Macros to format references to various kinds of objects.

6.16.1 L^AT_EX objects

\pkg {<name>} Also useable for class names.

```
662 \providerobustcmd*{\pkg}[1]{\mbox{\textsf{#1}}}
```

\cs {<csname>} From ltxdoc.

```
663 \providerobustcmd*{\cs}[1]{\texttt{\char'\#1}}
```

\env {<name>}

```
664 \providerobustcmd*{\env}[1]{\mbox{\texttt{#1}}}
```

`\marg` $\{\langle argument \rangle\}$ From ltxdoc.

```
665 \providecommand\marg[1]{%
666   {\ttfamily\char'\{\}\meta{#1}\ttfamily\char'\}}
```

`\oarg` $[\langle argument \rangle]$ From ltxdoc.

```
667 \providecommand\oarg[1]{%
668   {\ttfamily[]\meta{#1}\ttfamily[]}}
```

`\parg` $(\langle argument \rangle)$ From ltxdoc.

```
669 \providecommand\parg[1]{%
670   {\ttfamily{}\meta{#1}\ttfamily{}}}
```

`\ctr` $\{\langle name \rangle\}$

```
671 \providerobustcmd*{\ctr}[1]{\mbox{\texttt{#1}}}
```

`\bool` $\{\langle name \rangle\}$

```
672 \providerobustcmd*{\bool}[1]{\mbox{\texttt{#1}}}
```

`\optn` $\{\langle name \rangle\}$

```
673 \providerobustcmd*{\optn}[1]{\mbox{\texttt{#1}}}
```

`\TOC`

```
674 \providerobustcmd*{\TOC}{\acro{TOC}}
```

`\LOF`

```
675 \providerobustcmd*{\LOF}{\acro{LOF}}
```

`\LOT`

```
676 \providerobustcmd*{\LOT}{\acro{LOT}}
```

6.16.2 Programs and commands

`\cmds` $\{\langle \textit{commands to print} \rangle\}$ No processing is provided for special characters.

```
677 \providerobustcmd*{\cmds}[1]{\mbox{\textbf{\texttt{#1}}}}
```

`\procode` $\{\langle \textit{code to print} \rangle\}$ No processing is provided for special characters.

```
678 \providerobustcmd*{\procode}[1]{\mbox{\texttt{#1}}}
```

`\prog` $\{\langle \textit{program name} \rangle\}$ Underscores are allowed.

```
679 \newcommand*{\DTXD@prog}[1]{%
680   \mbox{\textsf{\textsl{\detokenize{#1}}}}%
681   \endgroup%
682 }
683
684 \providerobustcmd*{\prog}{%
685   \begingroup%
686   \catcode'\_ =12%
687   \DTXD@prog%
688 }
```

`\filenm` $\{\langle \textit{file name} \rangle\}$ Underscores are allowed.

```
689 \newcommand*{\DTXD@filenm}[1]{%
690   \mbox{\texttt{\detokenize{#1}}}%
691   \endgroup%
692 }
693
694 \providerobustcmd*{\filenm}{%
695   \begingroup%
696   \catcode'\_ =12%
697   \DTXD@filenm%
698 }
```

`\UI` General user-interface text.

```
699 \providerobustcmd*{\UI}[1]{\textbf{\textsf{#1}}}
```

6.16.3 File types

`\ODT`

```
700 \providerobustcmd*{\ODT}{\acro{ODT}}
```

\SVG

701 \providerobustcmd*{\SVG}{\acro{SVG}}

\PNG

702 \providerobustcmd*{\PNG}{\acro{PNG}}

\GIF

703 \providerobustcmd*{\GIF}{\acro{GIF}}

\JPG

704 \providerobustcmd*{\JPG}{\acro{JPG}}

\EPS

705 \providerobustcmd*{\EPS}{\acro{EPS}}

\PDF

706 \providerobustcmd*{\PDF}{\acro{PDF}}

\DVI

707 \providerobustcmd*{\DVI}{\acro{DVI}}

6.16.4 Internet

\UTF

708 \providerobustcmd*{\UTF}{\acro{UTF}}

\URL

709 \providerobustcmd*{\URL}{\acro{URL}}

\element {<name>}

710 \providerobustcmd*{\element}[1]{\texttt{<#1>}}

`\attribute` $\{\langle name \rangle\}$

`\attrib` $\{\langle name \rangle\}$

Each of these is “provided”, and any prior meaning will be unchanged. In particular, LuaTeX uses `\attribute`, so its meaning is unchanged if using LuaTeX.

```
711 \providerobustcmd*{\attrib}[1]{\mbox{\texttt{#1}}}
```

```
712
```

```
713 \providerobustcmd*{\attribute}[1]{\mbox{\texttt{#1}}}
```

`\HTML`

```
714 \providerobustcmd*{\HTML}{\acro{HTML}}
```

`\HTMLfive`

```
715 \providerobustcmd*{\HTMLfive}{\HTML\textsc{5}}
```

`\CSS`

```
716 \providerobustcmd*{\CSS}{\acro{CSS}}
```

`\CSSthree`

```
717 \providerobustcmd*{\CSSthree}{\CSS\textsc{3}}
```

`\EPUB`

```
718 \providerobustcmd*{\EPUB}{\acro{EPUB}}
```

6.16.5 Specific programs

`\tikz`

```
719 \providerobustcmd*{\tikz}{Ti\textit{k}z}
```

`\CTAN`

```
720 \providerobustcmd*{\CTAN}{\acro{CTAN}}
```

\TDS

721 \providerobustcmd*{\TDS}{\acro{TDS}}

\MathML

722 \providerobustcmd*{\MathML}{Math\acro{ML}}

\MathJax

723 \providerobustcmd*{\MathJax}{\brand{MathJax}}

6.16.6 Acronyms, brand names, trademarks

\brand {*<name>*}

724 \providerobustcmd*{\brand}[1]{\textsc{#1}}

\acro {*<acronym>*}

725 \providerobustcmd*{\acro}[1]{\textsc{\lowercase{#1}}}

\supregistered Superscript trademark symbol.

726 \providerobustcmd*{\supregistered}{\textregistered}

6.17 Logos

\dviTeX DVI T_EX

727 \providerobustcmd*{\dviTeX}{\mbox{\DVI\, \TeX}}

\dviLaTeX DVI L^AT_EX

728 \providerobustcmd*{\dviLaTeX}{\mbox{\DVI\, \LaTeX}}

\pdfTeX PDF T_EX

729 \providerobustcmd*{\pdfTeX}{\mbox{\PDF\, \TeX}}

`\pdfLaTeX` **PDF^LTeX**

```
730 \providerobustcmd*{\pdfLaTeX}{\mbox{\PDF\, \LaTeX}}
```

`\LuaTeX` **LuaTeX**

```
731 \providerobustcmd*{\LuaTeX}{\mbox{Lua\TeX}}
```

`\LuaLaTeX` **Lua^LTeX**

```
732 \providerobustcmd*{\LuaLaTeX}{\mbox{Lua\LaTeX}}
```

`\XeTeX` **X_ETeX, X_E^LTeX**

`\XeLaTeX`

```
733 \providerobustcmd*{\XeTeXrevE}
734   {\hspace{-.1667em}\raisebox{-.5ex}{E}\hspace{-.125em}}
735
736 \AtBeginDocument{
737   \ifpackageloaded{graphics}{
738     \renewrobustcmd*{\XeTeXrevE}
739       {\hspace{-.1667em}\raisebox{-.5ex}{\reflectbox{E}}\hspace{-.125em}}
740   }{}
741 }
742
743 \providerobustcmd*{\XeTeX}{\mbox{X\XeTeXrevE\TeX}}
744 \providerobustcmd*{\XeLaTeX}{\mbox{X\XeTeXrevE\LaTeX}}
```

`\AmS` **A_MS**

```
745 \providerobustcmd*{\AmS}{%
746   \leavevmode\hbox{$\mathcal A\kern-.2em\lower.376ex%
747   \hbox{$\mathcal M$}\kern-.2em\mathcal S$}%
748 }
```

`\LyX` **LyX**

```
749 \providerobustcmd*{\LyX}{\textsf{LyX}}
```

`\BibTeX` **BiBTeX**

```
750 \providerobustcmd*{\BibTeX}{\mbox{B\textsc{ib}\TeX}}
```

`\MakeIndex` ***MakeIndex***

```
751 \providerobustcmd*{\MakeIndex}{\prog{MakeIndex}}
```


`\ConTeXt` `ConTeXt`

```
752 \providerobustcmd*{\ConTeXt}{\mbox{ConTeXt}}
```

`\MiKTeX` `MiKTeX`

```
753 \providerobustcmd*{\MiKTeX}{\mbox{MiKTeX}}
```

6.18 Dashes and slashes

`\thinspace` A breakable thin skip.

```
754 \DeclareRobustCommand{\thinspace}{\hspace 0.16667em\relax}
```

`\endash` An endash: –

```
755 \def\endash{-}
```

`\emdash` An emdash: —

```
756 \def\emdash{-}
```

`\thinbrspace` A thin space which allows a line break.

```
757 \newcommand{\thinbrspace}{%
758   \hspace{.16667em}\penalty\exhyphenpenalty\hspace{0pt}%
759 }
```

`\thinthinbrspace` A thin space which allows a line break.

```
760 \newcommand{\thinthinbrspace}{%
761   \hspace{.08333em}\penalty\exhyphenpenalty\hspace{0pt}%
762 }
```

`\Dash` An unbreakable thin space, emdash, and breakable thin space.

```
763 \newrobustcmd{\Dash}{\unskip\thinspace\emdash\thinbrspace}
```

`\dash` An unbreakable thin space, endash, and breakable thin space.

```
764 \newrobustcmd{\dash}{\unskip\thinspace\endash\thinbrspace}
```

`\Slash` An unbreakable very thin space, a slash, and a breakable thin space.

765 `\newrobustcmd{\Slash}{\unskip\hspace{.08333em}/\thinthinbrspace}`

7 Compiling dtxdescribe

To compile the dtxdescribe package:

Enter ⇒ **pdflatex dtxdescribe.ins**

To compile the dtxdescribe documentation

Enter ⇒ **pdflatex dtxdescribe.dtx**

(Several times)

Enter ⇒ **makeindex -s gglo.ist -o dtxdescribe.gls dtxdescribe.glo**

Enter ⇒ **makeindex -s gind.ist dtxdescribe**

Enter ⇒ **pdflatex dtxdescribe.dtx**

(Several times)

Change History and Index

Change History

v0.10		\fqquad: Added.	56
General: 2016/12/08 Initial ver	1	\fqquad: Added.	56
v0.11		\fquad: Added.	56
\DTXD@cmdmargintagindex: Index		\margintag: Uses \margintacolor.	52
tag no longer plural.	41	\margintagcolor: Added.	52
\DTXD@index: Index tag no longer		\userentry: Added.	58
plural.	39	\userentryname: Added.	57
\watchout: Changed to \raggedleft. . . .	52	\watchoutcolor: Added.	52
General: 2018/03/30	1	General: 2019/01/11	1
v1.00		Added formatted objects.	58
\DTXD@cmdmargintagindex: Sans tag		Added logos.	63
font.	40	Cut file name changed to	
\DTXD@filemarginparindex: Fix: File		dtxexample_cut.tex	53
class.	43	UIdisplay: Added.	57
\DTXD@printtype: Sans tag font. . . .	37	docsidebar: Added.	58
\DescribeClass: Fix: Allow		noindenvironment: Added.	56
underscore.	45	noindmacro: Added.	56
\DescribeDefault: Added.	48	sourcedisplay: Added.	57
\DescribeDefaultcolor: Added.	48	sourceverb: Added.	57
\DescribeMacro: Sans tag font.	42	fsourceverb: Added.	57
\DescribePackage: Fix: Allow		v1.01	
underscore.	45	\DTXD@cmdmargintagindex: Put	
\ItemDescribeArgument: Added.	49	margin tag class in brackets.	40
\ItemDescribeBoolean: Added.	49	\DTXD@margintag: Put margin tag	
\ItemDescribeClass: Added.	50	class in brackets.	38
\ItemDescribeCommand: Added.	51	\DescribeMacro: Put margin tag	
\ItemDescribeCounter: Added.	49	class in brackets.	42
\ItemDescribeEnv: Added.	49	\DescribeObject: Put margin tag	
\ItemDescribeFile: Added.	51	class in brackets.	46
\ItemDescribeKey: Added.	50	\DescribeOther: Put margin tag	
\ItemDescribeLength: Added.	49	class in brackets.	47
\ItemDescribeMacro: Added.	48	General: 2019/03/22	1
\ItemDescribeObject: Added.	51	Sanitize PDF bookmarks.	34
\ItemDescribeOption: Added.	50	UIdisplay: Reduced width.	57
\ItemDescribeOther: Added.	52	sourcedisplay: Reduced width.	57
\ItemDescribePackage: Added.	50	v1.02	
\ItemDescribeProgram: Added.	51	\DescribeEnv: Fix if not ltxdoc class.	43
\dtxexamplecodename: Added.	53	\DescribeMacro: Fix if not ltxdoc	
\dtxexampleresultname: Added.	53	class.	42

\DescribeObject: \raggedleft		\DescribeMacro: Reduce hbox	
margin par.	46	warnings.	42
Added \ignorespaces.	46	\DescribeObject: Reduce hbox	
\DescribeOther: \raggedleft		warnings.	46
margin par.	47	\DescribeOther: Reduce hbox	
Added \ignorespaces.	47	warnings.	47
\PrintEnvName: Fix if not ltxdoc		\ItemDescribeHook: Added.	50
class.	37	\MathJax: Added.	63
\usage: Fix if not ltxdoc class.	38	\attrib: For LuaTeX.	62
General: 2019/07/16	1	\cs: Provided.	58
Fix if not doc package.	36	\dviLaTeX: Added.	63
Fix if not hyperref package.	34	\dviTeX: Added.	63
Fix if not ltxdoc class.	33, 36	\marg: Provided.	59
DTXD@gobble: Fix if not doc package.	35	\oarg: Provided.	59
dtexample: Fix if not doc package.	54	\parg: Provided.	59
v1.03		\pdfLaTeX: Added.	64
\DTXD@cmdmargintagindex: Reduce		\pdfTeX: Added.	63
hbox warnings.	41	\shownesting: Added.	53
\DTXD@margintag: Reduce hbox		General: 2022/02/01	1
warnings.	38	No longer requires xifthen.	33
\DescribeHook: Added.	46		

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		B	
[H] (argument) [figure]	24	Before myenvironment	25
[M] (argument) [figure]	24	\BibTeX	17, <u>750</u>
[\mymacro]:		[bigfiles]:	
bold (argument)	24	another_big_file.txt (file)	22
		really_big_file.txt (file)	22
A		bold (argument) [\mymacro]	24
\acro	16, <u>725</u>	\bool	15, <u>672</u>
Additional Item [otherclass]	25	boolean:	
\AmS	17, <u>745</u>	[descexamples]:	
another_big_file.txt (file) [bigfiles] .	22	booleanname	26
argument:		[examples]:	
[\mymacro]:		sampleboolean	21
bold	24	booleanname (boolean) [descexamples] ..	26
[figure]:		\brand	16, <u>724</u>
[H]	24		
[M]	24	C	
\attrib	16, <u>711</u>	calc (package)	33
\attribute	16, <u>711</u>	caption (package)	33

<code>\captionsetup</code>	<code>\DescribeArgument</code>	9, 327
<code>[dtxexample]</code>	<code>\DescribeBoolean</code>	9, 329
<code>[photograph]</code>	<code>\DescribeClass</code>	9, 322
class:	<code>\DescribeCommand</code>	10, 310
<code>[descexamples]:</code>	<code>\DescribeCounter</code>	9, 333
<code>class_name</code>	<code>\DescribeDefault</code>	11, 401
<code>[examples]:</code>	<code>\DescribeDefaultcolor</code>	11, 400
<code>sample_class</code>	<code>\DescribeEnv</code>	8, 282
<code>class_name (class) [descexamples]</code> . . .	<code>\DescribeFile</code>	10, 298
<code>\cmds</code>	<code>\DescribeHook</code>	9, 335
<code>\cnameref</code>	<code>\DescribeKey</code>	9, 337
<code>[photograph]</code>	<code>\DescribeLength</code>	9, 331
<code>[color]:</code>	<code>\DescribeMacro</code>	8, 245
<code>DTXD@examplerulecolor</code>	<code>\DescribeObject</code>	10, 339
<code>othercolor</code>	<code>\DescribeOption</code>	10, 325
<code>somecolor</code>	<code>\DescribeOther</code>	10, 374
command:	<code>\DescribePackage</code>	9, 316
<code>[descexamples]:</code>	<code>\DescribeProgram</code>	10, 304
<code>command_name</code>	<code>docsidebar (environment)</code>	14, 649
<code>OS_command</code>	<code>\DTX@filename</code>	285
<code>command_name (command) [descexam-</code>	<code>\DTXD@cmdmargintagindex</code>	214
<code>ples]</code>	<code>\DTXD@DescribeClass</code>	319
<code>\ConTeXt</code>	<code>\DTXD@DescribeCommand</code>	307
counter:	<code>\DTXD@DescribeFile</code>	295
<code>[examples]:</code>	<code>\DTXD@DescribePackage</code>	313
<code>samplecounter</code>	<code>\DTXD@DescribeProgram</code>	301
<code>\crefname</code>	<code>DTXD@examplerulecolor [color]</code>	53
<code>[dtxexample]</code>	<code>\DTXD@filemarginparindex</code>	286
<code>\cs</code>	<code>\DTXD@gobble</code>	84
<code>\CSS</code>	<code>\DTXD@index</code>	177
<code>\CSSthree</code>	<code>\DTXD@macroname</code>	208
<code>\CTAN</code>	<code>\DTXD@margintag</code>	161
<code>\ctr</code>	<code>\DTXD@margintagindex</code>	203
	<code>\DTXD@origwindex</code>	160
	<code>\DTXD@printtype</code>	156
	<code>\DTXD@verbatimcmd</code>	211
	<code>[dtxexample]:</code>	
	<code>\captionsetup</code>	55
	<code>\crefname</code>	55
	<code>\DeclareFloatingEnvironment</code> . . .	55
	<code>dtxexample (environment)</code>	12, 549
	<code>dtxexample_cut.tex (file)</code>	53
	<code>\dtxexamplecodename</code>	13, 547
	<code>\dtxexampleresultname</code>	13, 548
	<code>\DVI</code>	15, 707
	<code>\dviLaTeX</code>	17, 728
	<code>\dviTeX</code>	17, 727
	E	
	<code>\element</code>	16, 710
D		
<code>\Dash</code>		17, 763
<code>\dash</code>		17, 764
<code>\DeclareFloatingEnvironment</code>		
<code>[dtxexample]</code>		55
<code>[photograph]</code>		19
<code>[descexamples]:</code>		
<code>booleanname (boolean)</code>		26
<code>class_name (class)</code>		26
<code>command_name (command)</code>		26
<code>file_name (file)</code>		26
<code>keyname (key)</code>		26
<code>\lengthname (length)</code>		26
<code>\macroname</code>		26
<code>package_name (package)</code>		26
<code>program_name (program)</code>		26

program:		T	
[descexamples]:		\TDS	16, 721
program_name	26	\thinbrspace	17, 757
program_name	22	\thinspace	17, 754
program_name (program)	22	\thinthinbrspace	17, 760
program_name (program) [descexamples]	26	\tikz	16, 719
		\TOC	15, 674
	R		
randomobject	25		
really_big_file.txt (file) [bigfiles] ..	22	U	
		\UI	15, 699
		UIDisplay (environment)	14, 629
	S	\URL	16, 709
sample_class (class) [examples]	22	\usage	158
sampleboolean (boolean) [examples] ..	21	\userentry	14, 641
samplecounter (counter) [examples] ..	21	\userentryname	14, 640
samplekey (key) [examples]	23	\UTF	16, 708
sampleoption (option) [examples]	22		
samplepackage (package) [examples] ..	22	W	
sampletwokey (key) [examples]	23	\warningsign	76
secondkey (key) [groupofkeys]	23	\watchout	12, 522
\shownesting	11, 531	\watchoutcolor	12, 521
\Slash	18, 765		
somecolor [color]	25	X	
sourcedisplay (environment) ...	14, 618	xcolor (package)	33
sourceverb (environment)	13, 612	\XeLaTeX	17, 733
splitidx (package)	33	\XeTeX	17, 733
\supregistered	16, 726	xparse (package)	33
\SVG	15, 701	xstring (package)	33