The LaTeX dtxdescribe Package

v0.10 — 2016/12/08

bd@BDTechConcepts.com

Describe additional object types in `dtx` source files.

Abstract

The doc package includes tools for describing macros and environments in LaTeX source `dtx` format. The dtxdescribe package adds additional tools for describing booleans, lengths, counters, keys, packages, classes, options, files, commands, arguments, and other objects.

Each 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.

Descriptions are best accompanied by examples, so the environment `dtxexample` is provided. Contents are displayed verbatim along with a caption and cross-referencing. They are then `\input` and executed, and the result is shown.

Contents

List of Examples

List of Figures

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 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.

Too much!

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 `\DescribeBoolean`, `\DescribeLength`, `\DescribeCounter`, and related macros, similar to the already-familiar `\DescribeMacro` and `\DescribeEnv`.

Optional "classes" may be assigned to the objects being described, including the new verisons 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` 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 LaTeX 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 Pre-existing Macros

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

The pre-existing macro from the doc is redefined to allow hyperlinked index entries and 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 10 for examples.

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

The pre-existing macro from the doc is redefined to allow hyperlinked index entries, and also to place an 'Env' tag in front of the name in the margin. See example 2 on page 11.

3.2 Macro Arguments

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

`\marg` `{`⟨*text*⟩`}`

Shows a mandatory argument for a macro or environment.

The results looks like `{`⟨*mandatory*⟩`}`.

`\oarg` `{`⟨*text*⟩`}`

Shows an optional argument for a macro or environment.

The results looks like `[`⟨*optional*⟩`]`.

`\parg` `{`⟨*text*⟩`}`

Used for "picture" arguments, such as coordinates.

The result looks like `(`⟨*coordinate*⟩`)`.

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

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 15.

3.3 Common LaTeX Elements

See example 4 on page 12.

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

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

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

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

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

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

`\DescribeKey` `[⟨class⟩] {⟨name⟩}`

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 14.

3.4 References to External Packages and Classes

`\DescribePackage` `[⟨class⟩] {⟨name⟩}`

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

`\DescribeClass` `[⟨class⟩] {⟨name⟩}`

Describes a LaTeX class. Given a 'Cls' tag in the margin and index.

`\DescribeOption` `[⟨class⟩] {⟨name⟩}`

Describes a LaTeX package or class option. Given an 'Opt' tag in the margin and index.

3.5 Files, Programs, and 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 16.

`\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 16.

3.7 Additional Tags

`\margintag` `{⟨text⟩}`

Creates a colored margin tag. May be used to identify the topic of a paragraph or the destination of an arbitrary index entry.

`\watchout` `[⟨text⟩]`

Creates a red margin tag with a warning sign and optional text. May be used to warn the reader of special instructions, etc.

3.8 `dtxexample` Environment

Env `dtxexample` `* [⟨Notes/cross-references⟩] {⟨caption & label⟩}`

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 12 for a demonstration of how `dtxexample` works.

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:

`\mymacro` `[`⟨*optional*⟩`]` `{`⟨*mandatory*⟩`}` A typical macro definition.

photograph Create a photograph float.
`\DeclareFloatingEnvironment`

photograph `\captionsetup` Caption settings for a photograph float.

photograph `\cnameref` cleveref 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:

Env `myenvironment` {⟨*argument*⟩} Short description.

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:

Env `kindofenvironment` `otherenvironment` [⟨*opt args*⟩] (⟨*coordinates*⟩) A description.

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]{sampleclass}
    About a \LaTeX\ class.

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

Result:

Pkg examples **samplepackage** About a LaTeX package.

Cls examples **sampleclass** About a LaTeX 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 **really_big_file.txt** Some description.

File **another_big_file.txt** Some description.

File **lone_file.txt** Some description.

Cmd **OS_command** An operating-system command.

Prog **program_name** An operating-system program.

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.
```

Result:

color `somecolor` The color of something.

color `othercolor` The other color.

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.

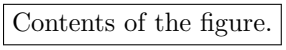

Figure 1: A Figure

Example 12: `dtxexample`

Code:

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

Result:

See fig. 1

Example 12, typeset above, was created with the following code:

```
\begin{dtxexample}[See \cref{fig:afigure}]
    {\env{dtxexample}\label{ex:dtxexample}}
\begin{figure}
    \centering\fbox{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 12.
2. The code was displayed verbatim.
3. The code was written to the file `ex_cut.tex`.
4. The code was `\input` from `ex_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.

5 Usage Notes

Placement of `\Describe` macros: Typically LaTeX 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 `\Describe`, `\index`, `\margintag`, and `\watchout` macros together, care must be taken to avoid extra space in the printed text where these macros occur. Try to place the first one directly connected to a word, and the others may follow on the next line if necessary.

```
text text text\margintag{A comment.}\index{An entry}
\index{Another entry}
more inline text text text
```

`\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 supressed (there is no margin in a float), but the index entries are still created.

[1] Future 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 **etoolbox** v2.6 or later for `\BeforeBeginEnvironment`, `\AfterEndEnvironment`

```
\RequirePackage{etoolbox}[2011/01/03]%
```

Pkg **xparse** Used for the examples.

```
\RequirePackage{xparse}
```

Pkg **xifthen** Used for the examples.

```
\RequirePackage{xifthen}
```

Pkg **xcolor** Used for the examples.

```
\RequirePackage{xcolor}
\definecolor{myurlcolor}{rgb}{0,0,.7}
\definecolor{mylinkcolor}{rgb}{.7,0,0}
```

Pkg **caption** Used for the examples.

```
\RequirePackage{caption}
```

Pkg **newfloat** Used for the examples.

```
\RequirePackage{newfloat}
```

Pkg **fancyvrb** Used for the examples.

```
\RequirePackage{fancyvrb}
```

Pkg **xstring** Used for `\StrSubstitute` for `\DescribeFile`.

```
\RequirePackage{xstring}
```

Pkg **pict2e**

```
\RequirePackage{pict2e}
\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.

```
\newcommand*{\warningsign}{%
\begin{picture}(10,9)
\put(4,1){\scriptsize!}
\put(0,0){\line(500,866){5}}
\put(10,0){\line(-500,866){5}}
\put(0,0){\line(1,0){10}}
\end{picture}
}
```

6.2 Support Macros

```
\renewcommand*{\PrintEnvName}[1]
{\strut{\scriptsize{}Env}\quad\MacroFont#1\ }
```

`\DTXD@printtype` `{`⟨*text*⟩`}`

Used to print the object class in the margin:

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

`\usage` `{`⟨*text*⟩`}`

Allow hyperlinks in the "usage" index entries:

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

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

```
\let\DTXD@origwrindex\@wrindex
```

`\DTXD@margintag` `{`⟨*class*⟩`}` `{`⟨*name*⟩`}` `{`⟨*margin tag*⟩`}`

Creates the margin tag for the object being described.

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

```
\newcommand*{\DTXD@margintag}[3]{%
\@ifundefined{@captype}{% not float?
\leavevmode%
\marginpar{%
\DTXD@printtype{%
#3% margintag
```

```
\ifblank{#1}{}{ \texttt{#1}}% class
}% Desc@Type
\texttt{#2}% name
}% marginpar
}{}% not float?
}
```

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

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.

```
\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.

```
\@bsphack%
\begingroup%
\DTXD@origwrindex{%
```

Index by name:

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

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

Index by tag and class:

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

```
\begingroup%
\DTXD@origwrindex{%
#4s:\levelchar% index tag
\ifblank{#1}{}{[#1]:\levelchar}% class
#2\actualchar{\protect\ttfamily#2}% name
\encapchar #5}%
```

Possibly index by class and name:

```
\ifblank{#1}{}{% class given
\begingroup%
\DTXD@origwrindex{%
#1\actualchar[#1]:\levelchar% class
#2\actualchar{\protect\ttfamily#2} % name
(#4)% index tag
\encapchar #5}%
}% class given
% \@esphack%
\@esphack%
\ignorespaces%
}
```

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

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

```
\newcommand*{\DTXD@margintagindex}[5]{%
% \@bsphack%
```

The margin tag and the name:

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

The index entries:

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

`\DTXD@macroname` `{`⟨*control sequence*⟩`}`

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

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

`\DTXD@verbatimcmd` `{`⟨`\name`⟩`}`

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

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

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

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

```
\newcommand*{\DTXD@cmdmargintagindex}[5]{%
\@bsphack%
```

Create a margin tag with the name of the macro:

```
\@ifundefined{@captype}{% not float?
\leavevmode%
\marginpar{%
\DTXD@printtype{%
#3% margin tag
\ifblank{#1}{}{ \texttt{#1}}% class
}% Desc@Type
\cmd{#2}% name
}% marginpar
}{}% 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`

```
\begingroup%
\DTXD@origwrindex{%
\ifblank{#1}{}{#1\actualchar[#1]:\levelchar}% class
\DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2} % name
(#4)% index tag
\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

```
\begingroup%
\DTXD@origwrindex{%
#4s:\levelchar% index tag
\ifblank{#1}{}{[#1]:\levelchar}% class
\DTXD@verbatimcmd{#2}% name
\encapchar #5}%
\@esphack%
\ignorespaces%
}

```

6.3 Pre-existing Macros

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

Redefined to allow hyperlinked index entries and an optional class:

```
\renewcommand*{\DescribeMacro}[2][]{%
\@bsphack%
```

Create the margin tag with the macro's name:

```
\@ifundefined{@captype}{% not float?
\leavevmode%
\marginpar{%
\raggedleft%
\ifblank{#1}{}{{\scriptsize#1} }% class
\cmd{#2}% name
}% marginpar
}{}% 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

```
\begingroup%
\DTXD@origwrindex{%
\DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2}% name
\ifblank{#1}{}{\levelchar[#1]}% class
\encapchar usage}%
```

Only if a class was given:

```
\ifthenelse{\isempty{#1}}%
```

```
{}% no class
{% class given
% Again, and prepend the class:
%
% Write class=(class):>name=csname\verb+|usage+
%    \begin{macrocode}
\begingroup%
\DTXD@origwrindex{%
#1\actualchar[#1]:\levelchar%
\DTXD@macroname{#2}\actualchar\DTXD@verbatimcmd{#2}%
\encapchar usage}%
}% class given
\@esphack%
\ignorespaces%
}
```

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

Redefined to allow hyperlinked index entries:

```
\renewcommand*{\DescribeEnv}[2][]
{\DTXD@margintagindex{#1}{#2}{Env}{environment}{usage}}
```

6.4 New Describe Macros

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

```
\newcommand*{\DTXD@filename}{}
```

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

The name may have underscores.

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

Create a detokenized version of the filename...

```
\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:

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

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

```
\DTXD@margintag{}{#2}{#3}%
```

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

```
\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:

```
\endgroup%
\ignorespaces%
}
```

`\DTXD@DescribeFile` `[`⟨*class*⟩`] {`⟨*name*⟩`}`

The name may have underscores.

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

`\DescribeFile` `{`⟨*name*⟩`}`

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

```
\newcommand*{\DescribeFile}{%
\begingroup\catcode`\_=12 \DTXD@DescribeFile%
}
```

`\DTXD@DescribeProgram` `[`⟨*class*⟩`] {`⟨*name*⟩`}`

The name may have underscores.

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

`\DescribeProgram` `{`⟨*name*⟩`}`

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

```
\newcommand*{\DescribeProgram}{%
```

```
\begingroup\catcode`\_=12 \DTXD@DescribeProgram%
}
```

`\DTXD@DescribeCommand` `[`⟨*class*⟩`] {`⟨*name*⟩`}`

The name may have underscores.

```
\newcommand*{\DTXD@DescribeCommand}[2][]{%
\DTXD@filemarginparindex{#1}{#2}{Cmd}{command}{usage}%
}
```

`\DescribeCommand` `{`⟨*name*⟩`}`

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

```
\newcommand*{\DescribeCommand}{%
\begingroup\catcode`\_=12 \DTXD@DescribeCommand%
}
```

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

```
\newcommand*{\DescribePackage}[2][]
{\DTXD@margintagindex{#1}{#2}{Pkg}{package}{usage}}
```

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

```
\newcommand*{\DescribeClass}[2][]
{\DTXD@margintagindex{#1}{#2}{Cls}{class}{usage}}
```

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

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

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

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

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

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

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

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

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

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

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

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

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

```
\newcommand*{\DescribeKey}[2][]
{\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`.

```
\newcommand*{\DescribeObject}[2][]{%
\@ifundefined{@captype}{% not float?
\@bsphack%
\leavevmode\marginpar{\raggedleft{\scriptsize#1} \texttt{#2}}%
}{}% not float?
\ifthenelse{\isempty{#1}}
{\begingroup%
\DTXD@origwrindex{%
#2\actualchar{\protect\ttfamily#2}%
\encapchar usage%
}%
}%
{%
\begingroup%
\DTXD@origwrindex{%
#2\actualchar{\protect\ttfamily#2} [#1]%
\encapchar usage%
}%
\begingroup%
\DTXD@origwrindex{%
#1\actualchar[#1]:\levelchar#2\actualchar{\protect\ttfamily#2}%
\encapchar usage%
}%
}%
\@esphack%
% \ignorespaces%
```

```
207 }
```

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

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

```
208 \newcommand*{\DescribeOther}[2][]{%
209 \@ifundefined{@captype}{% not float?
210 \@bsphack%
211 \leavevmode\marginpar{\raggedleft{\scriptsize#1} #2}%
212 }{}% not float?
213 \ifthenelse{\isempty{#1}}
214 {%
215 \begingroup%
216 \DTXD@origwrindex{#2\encapchar usage}%
217 }%
218 {%
219 \begingroup%
220 \DTXD@origwrindex{#2 [#1]\encapchar usage}%
221 \begingroup%
222 \DTXD@origwrindex{#1\actualchar[#1]:\levelchar#2\encapchar usage}%
223 }%
224 \@esphack%
225 % \ignorespaces%
226 }
```

6.5 New Margin Tags

`\margintag` `{`⟨*text*⟩`}`

Prints a colored margin tag.

```
227 \newcommand{\margintag}[1]{%
228 \@ifundefined{@captype}{% not float?
229 \marginpar{\raggedleft\textcolor{blue!70!black}{#1}}%
230 \ignorespaces%
231 }{}% not float?
232 }
```

`\watchout` `[`⟨*text*⟩`]`

Prints a warning sign and optional text.

```
233 \newcommand{\watchout}[1][]{%
234 \@ifundefined{@captype}{% not float?
```

```
% \@bsphack%
\marginpar{\hspace*{\fill}%
\textcolor{red!50!black}{\warningsign\normalsize\quad#1}}%
% \@esphack%
\ignorespaces%
}{}% not float?
}
```

6.6 The `dtxexample` Environment

Also see example 12 on page page 17.

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

color `DTXD@examplerulecolor` The color of the middle rule in the dtxexample.

```
\definecolor{DTXD@examplerulecolor}{rgb}{.9,.9,.9}
```

Env `dtxexample` * [⟨*notes/cross-references*⟩] {⟨*caption & label*⟩}

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.

```
\NewDocumentEnvironment{dtxexample}{s +O{} m}
{% start dtxexample
```

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

```
\VerbatimOut[gobble=2,tabsize=4]{ex_cut.tex}%
}% start dtxexample
```

When the environment closes:

```
{% end dtxexample
```

Finish the verbatim output:

```
\endVerbatimOut
\par
\addvspace{\bigskipamount}
```

If unstarred, typeset the example in a minipage:

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

Emulated a float of type “example”:

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

Typeset the contents as verbatim:

```
\textcolor{DTXD@examplerulecolor}{\smallskip\hrule}
\smallskip
{\scriptsize\itshape Code:}
\VerbatimInput[tabsize=4]{ex_cut.tex}
\unskip
\textcolor{DTXD@examplerulecolor}{\hrule}
\smallskip
{\scriptsize\itshape Result:}

```

Possible add the optional cross-references or notes:

```
\ifstrempty{#2}
{}
{{\itshape\small #2}}
```

If unstarred, close the `\minipage`.

```
\IfBooleanTF{#1}{}{\endminipage}%
} % end dtxexample
```

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

```
\AfterEndEnvironment{dtxexample}
{%
```

Execute the code:

```
\par\unskip\input{ex_cut.tex}%
```

Closing rule::

```
\medskip\hrule%
}
```

dtxexample A new float type for the examples.

\DeclareFloatingEnvironment

```
\DeclareFloatingEnvironment[
fileext=lox,
listname={List of Examples},
name=Example,
placement=hbp
]{dtxdexample}
```

dtxexample \captionsetup Caption setup for the examples.

```
\captionsetup*[dtxdexample]{
format=hang,
font=bf,
justification=raggedright,
singlelinecheck=false,
skip=0pt,
position=top,
}
```

dtxexample \crefname Name for cleveref.

```
\AtBeginDocument{
\@ifpackageloaded{cleveref}{\crefname{dtxdexample}{example}{examples}}{}
}
```

Change History and Index

Change History

v0.10
General: 2016/12/08 Initial ver . . 1

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.

E

F

G

I

K

L

M

N