

The `rgltxdoc` package*

Richard Grewe
r-g+tex@posteo.net

May 18, 2019

1 Introduction

This package combines several other packages and defines additional macros and environments for the purpose of documenting LaTeX code. The package mainly serves the purpose of combining the preferences used in the author's package documentations. However, others can use the package as well. Compatibility between versions cannot be guaranteed, however.

2 Basic Dependencies

Generally, the documentation can be compiled with pdfL^AT_EX and with LuaL^AT_EX. Other processors are untested. There is no assertion as to how close the pdfL^AT_EX and LuaL^AT_EX results are to each other.

```
1 \RequirePackage{ifluatex}
```

The `etoolbox` package is used to simplify some of the package's code.

```
2 \RequirePackage{etoolbox}
```

3 Documentation Input

The documentation is expected to be written in UTF-8 and in US-English language. If `babel` is already loaded, it will not be loaded again, though. This is to support packages that use other languages than English in examples and load `babel` accordingly.

```
3 \ifbool{luatex}{  
4   \RequirePackage[utf8]{luainputenc}  
5   \RequirePackage{polyglossia}  
6   \setmainlanguage[variant=american]{english}  
7 }{  
8   \RequirePackage[utf8]{inputenc}  
9   \@ifpackageloaded{babel}{}
```

*This document corresponds to `rgltxdoc` v1.2, dated 2019/05/04. The package is available online at <http://www.ctan.org/pkg/rgltxdoc> and <https://github.com/Ri-Ga/rgltxdoc>.

```

10   {\RequirePackage[english]{babel}}
11 }

```

4 General Appearance

Code in this section determines the general appearance of documentation text and is not specific to documenting L^AT_EX code.

4.1 Page Layout

For the page layout, A4 is used for the paper size. Border correction established wider left margins for typesetting long macro names. The DIV value is tuned to make the lines wide enough to support at least 72 characters in the package documentation code.

```

12 \RequirePackage[a4paper,twoside=false]{geometry}
13 \RequirePackage[DIV=9,BCOR=2.25cm]{typearea}

```

4.2 Fonts

For the font, Latin Modern is used. Particularly, a light version of the typewriter font is used, such that highlighting in listings is possible via a bold font series.

```

14 \ifbool{luatex}{
15   \RequirePackage{fontspec}
16   \setmainfont[SmallCapsFont={* Caps}]{Latin Modern Roman}
17   \setsansfont{Latin Modern Sans}
18   \setmonofont[Scale=MatchLowercase,
19               SmallCapsFont={Latin Modern Mono Caps}]
20     {Latin Modern Mono Light}
21 }{
22   \RequirePackage[T1]{fontenc}
23   \RequirePackage[lighttt]{lmodern}

```

With just the above code, a construct like `\cs{foo\meta{bar}}` for documenting parameter-dependent macro names fails due to missing fonts. The following two lines fix this. The first line ensures that the typewriter font is loaded (via an `\hbox` with typewriter text that is not actually displayed) and the second line declares the required font shape (see <https://tex.stackexchange.com/questions/234003/italic-font-in-lmodern-lighttt>).

```

24   \bgroup\setbox\z@\hbox{\ttfamily ignore}\egroup
25   \DeclareFontShape{T1}{lmtt}{m}{it}{<->sub*lmtt/m/sl}{}
26 }

```

Finally, `microtype` is used for small font improvements.

```

27 \RequirePackage{microtype}

```

We simplify quoting names through the `csquotes` package and register `"` to produce double opening/closing quotation marks.

```

28 \RequirePackage[autostyle=true]{csquotes}
29 \MakeOuterQuote{"}

```

4.3 Document Structure

For the most part, documentations are structured as usual, through a title as well as sections and sub-sections and so forth. The following two packages improve the possibilities for using lists in documentation and visually improve the index through a two-column layout.

```
30 \RequirePackage[inline]{enumitem}
31 \RequirePackage[columns=2]{idxlayout}
```

The `cleveref` and `varioref` packages shall be used for referencing structural entities, such as sections and figures. Hyperlinks are enabled through `hypdoc`.

```
32 \RequirePackage{varioref}
33 \RequirePackage{hypdoc}
34 \RequirePackage[capitalise,noabbrev,nameinlink]{cleveref}
```

5 Documenting Things

This package builds on the `doc` package for several documentation macros, such as `\marg`, `\oarg`, and `\meta`.

```
35 \RequirePackage{doc}
```

5.1 Macros and Environments

The main macros here, `\NiceDescribeMacro` and `\NiceDescribeEnv`, are references to `\DescribeMacro` and `\DescribeEnv` of the `doc` package, with which they share the purpose. The main difference is the appearance in that the “nice” macros include the argument list.

```
\NiceDescribeMacro [<idx>] {<macro>} {<parameters>}
\niceDescribeMacros {<n>} [<idx1>] {<macro1>} {<params1>} ... [<idxn>] {<macron>} {<paramsn>}
```

These macros produce a description header for a single macro or, respectively, for multiple macros. The above two lines are an example of such a description header, which is produced by the following code:

```
\NiceDescribeMacros{2}
{ \NiceDescribeMacro } { \oarg{idx} \marg{macro} \marg{parameters} }
{ \NiceDescribeMacros } { \marg{n}
  \oarg{idx$_1$} \marg{macro$_1$} \marg{params$_1$} \ldots
  \oarg{idx$_n$} \marg{macro$_n$} \marg{params$_n$} }
```

The arguments to the macro are described below:

<i><n></i>	This argument specifies the number of macros to be described.
<i><macro></i> , <i><macro₁></i> , ..., <i><macro_n></i>	These arguments specify the macros for which a description header shall be produced.
<i><parameters></i> , <i><params₁></i> , ..., <i><params_n></i>	These arguments take the sequence that specifies all optional and mandatory arguments of the respective <i><macro></i> . Typically, these would be sequences of <code>\oarg</code> and <code>\marg</code> instances.

$\langle idx \rangle$,	These optional arguments specify which index entries
$\langle idx_1 \rangle, \dots,$	shall be documented, if they differ from the respective $\langle macro \rangle$
$\langle idx_N \rangle$	parameters. This, for instance allows $\langle macro \rangle$ to be “\foo(*)”
	whereas the $\langle idx \rangle$ parameter could be “\foo, \foo*”.

```
\NiceDescribeEnv[\langle idx \rangle]{\langle environment \rangle}{\langle parameters \rangle}
\nNiceDescribeEnvs{\langle n \rangle}[\langle idx_1 \rangle]{\langle env_1 \rangle}{\langle params_1 \rangle}...[\langle idx_n \rangle]{\langle env_n \rangle}{\langle params_n \rangle}
```

These macros are the counterparts of `\NiceDescribeMacro` and, respectively, `\NiceDescribeMacros` when it comes to L^AT_EX environments. The $\langle environment \rangle$ (resp. $\langle env_1 \rangle$ to $\langle env_n \rangle$) parameters are the names of the respective environments. A usage example can be found in the implementation part of [Section 5.2](#) on page 8.

```
\NiceDescribeCounter[\langle idx \rangle]{\langle counter \rangle}{\langle qualifiers \rangle}
\nNiceDescribeCounters{\langle n \rangle}[\langle idx_1 \rangle]{\langle ctr_1 \rangle}{\langle qual_1 \rangle}...[\langle idx_n \rangle]{\langle ctr_n \rangle}{\langle qual_n \rangle}
```

These macros are analogous to the above macros, but aimed for documenting L^AT_EX counters.

```
\NiceDescribeKey[\langle idx \rangle]{\langle keyname \rangle}{\langle keyconfig \rangle}
\nNiceDescribeKeys{\langle n \rangle}[\langle idx_1 \rangle]{\langle name_1 \rangle}{\langle cfg_1 \rangle}...[\langle idx_n \rangle]{\langle name_n \rangle}{\langle cfg_n \rangle}
```

These macros are for documenting option keys, for instances of packages `xkeyval` or `pgfkeys`. The $\langle keyconfig \rangle$ is a key-value list in which the keys “vals”, “init”, and “def” can be used to specify the range of expected/permitted values, the initial (preset) value, and the default (if the key is provided without a value).

```
\NewNiceDescription{\langle type \rangle}{\langle efmt \rangle}{\langle afmt \rangle}{\langle icmd \rangle}
```

This macro is used internally for defining the above macros and can be used for defining new types of entity descriptions. The following table describes the arguments of the macro.

$\langle type \rangle$	The $\langle type \rangle$ argument is the name of the type of entities.
$\langle efmt \rangle$	The $\langle efmt \rangle$ argument is T _E X code that formats the entities in the margin. It can – and should – reference the positional parameter \#1, through which it is passed the name of the entities.
$\langle afmt \rangle$	The $\langle afmt \rangle$ argument is T _E X code that formats the arguments or qualifiers of the entities in the body of the documentation. Analogous to $\langle efmt \rangle$, also $\langle afmt \rangle$ receives the arguments/qualifiers through the positional parameter \#1.
$\langle icmd \rangle$	The $\langle icmd \rangle$ argument is T _E X code that adds a usage entry for the entity to the index. It takes one argument, through which $\langle icmd \rangle$ is passed the entity name.

A usage example for `\NewNiceDescription` can be found in the implementation below.

Implementation

<code>\NewNiceDescription</code>	<p>The <code>\NewNiceDescription{⟨type⟩}{⟨efmt⟩}{⟨afmt⟩}{⟨icmd⟩}</code> macro defines the <code>\NiceDescribe⟨type⟩</code> and <code>\NiceDescribe⟨type⟩s</code> macros and saves the <code>⟨efmt⟩</code> and <code>⟨afmt⟩</code> parameters for use by <code>\NiceDescribe⟨type⟩s</code>.</p> <pre> 36 \newcommand\NewNiceDescription[4]{% 37 \expandafter\newcommand\csname NiceDescribe#1\endcsname{% 38 \csname NiceDescribe#1s\endcsname{1}}}% 39 \expandafter\newcommand\csname NiceDescribe#1s\endcsname{% 40 \rgltxdoc@Desc 41 {\csuse{rgltxdoc@@efmt@#1}} 42 {\csuse{rgltxdoc@@afmt@#1}} 43 {#4}}% 44 \csdef{rgltxdoc@@efmt@#1}##1{#2}% 45 \csdef{rgltxdoc@@afmt@#1}##1{#3}}</pre>
<code>\NiceDescribeMacro</code> <code>\NiceDescribeMacros</code>	<p>Macro names are formatted detokenized through <code>\string</code>. Arguments are formatted as is. For the index, <code>\SpecialUsageIndex</code> of the <code>doc</code> package is used.</p> <pre> 46 \NewNiceDescription{Macro}{\string#1}{#1}{\SpecialUsageIndex}</pre>
<code>\NiceDescribeEnv</code> <code>\NiceDescribeEnvs</code>	<p>Environment names are formatted with a gray <code>\begin</code> and <code>\end</code>. The arguments of environments are formatted as is. For the index, <code>\SpecialEnvIndex</code> of the <code>doc</code> package is used.</p> <pre> 47 \NewNiceDescription{Env} 48 {\textcolor{gray}{\cs{begin}}\cmarg{#1}\ 49 \textcolor{gray}{\cs{end}}\cmarg{#1}} 50 {#1}{\SpecialEnvIndex}</pre>
<code>\NiceDescribeCounter</code> <code>\NiceDescribeCounters</code>	<p>Counter names are formatted as is. Arguments or qualifiers should usually not be present for counters, but if provided, they would be formatted as is. The index entry is produced through <code>\SpecialOtherIndex</code> (see its documentation below).</p> <pre> 51 \NewNiceDescription{Counter}{#1}{#1} 52 {\SpecialOtherIndex{counter}{counters}}</pre>
<code>\NiceDescribeKey</code> <code>\NiceDescribeKeys</code>	<p>Option-key names are formatted as is. Arguments are split into the range of values (vals), the initial value (init), and the default value (def – the value used when only the key but no value is specified to the key). These three can be set as keys. The index entry is produced through <code>\SpecialOtherIndex</code> (see its documentation below).</p> <pre> 53 \NewNiceDescription{Key}{#1\,\textrm{=}\,\null} 54 {\begingroup 55 \newcommand\vmeta[1]{\normalfont\meta{##1}}}% 56 \setkeys[rgltxdoc]{DescOpt}{#1}% 57 \setbox\z@\hbox{\qqad\let\rgltxdoc@@sep\empty 58 \rgltxdoc@opt{default}{\cmdrgltxdoc@DescOpt@def}}% 59 \rgltxdoc@opt{initially}{\cmdrgltxdoc@DescOpt@init}}% 60 \parbox[t]{\linewidth-\wd\z@}{% 61 \raggedright\cmdrgltxdoc@DescOpt@vals}% 62 \box\z@\endgroup} 63 {\SpecialOtherIndex{option-key}{option-keys}} 64 \usepackage{xkeyval,calc} 65 \define@key[rgltxdoc]{DescOpt}{vals}{%</pre>

```

66 \def\cmdrgltxdoc@DescOpt@vals{}\let\rgltxdoc@@sep\empty%
67 \def\do##1{%
68   \protected@eappto\cmdrgltxdoc@DescOpt@vals{\rgltxdoc@@sep\texttt{##1}}%
69   \def\rgltxdoc@@sep{, }%
70   \docsvlist{##1}}
71 \define@cmdkey[rgltxdoc]{DescOpt}{init}{}
72 \define@cmdkey[rgltxdoc]{DescOpt}{def}{}
73 \newcommand\rgltxdoc@opt[2]{\ifdef{#2}
74   {\rgltxdoc@@sep\textsl{#1: }\texttt{#2}}%
75   \def\rgltxdoc@@sep{, }}
76 {}

```

`\SpecialOtherIndex` The `\SpecialOtherIndex{<type>}{<types>}{<name>}` macro adds an index entry of the given `<type>` (with plural form `<types>`) and given `<name>`. The macro is a straightforward generalization of `\SpecialEnvIndex` (both from `hypdoc` and from `doc`).

```

77 \newcommand*\SpecialOtherIndex[3]{%
78   \@bsphack
79   \begingroup
80     \HD@target
81     \let\HDorg@encapchar\encapchar
82     \edef\encapchar usage{%
83       \HDorg@encapchar hdclindex{\the\c@HD@hypercount}{usage}%
84     }%
85     \rgltxdoc@nohyp@SpecialOtherIndex{#1}{#2}{#3}%
86   \endgroup
87   \@esphack}
88 \newcommand\rgltxdoc@nohyp@SpecialOtherIndex[3]{%
89   \index{#3\actualchar{\protect\ttfamily#3}
90     (#1)\encapchar usage}%
91   \index{#2:\levelchar#3\actualchar
92     {\protect\ttfamily#3}\encapchar usage}}

```

`\rgltxdoc@DescRec` The `\rgltxdoc@Desc{<efmt>}{<afmt>}{<icmd>}{<n>}[<idx>]{<entity>}{<args>}` macro formats a description header for `<n>` entities, of which the first are specified through `<idx>`, `<entity>`, and `<args>`. The margin parts are formatted through the `<efmt>{<entity>}` macro, the parts in the text body through the `<afmt>{<args>}` macro. The index entries are created through the `<icmd>{<idx>}` macro. In its implementation, `\rgltxdoc@Desc` builds on `\pbox` from the `pbox` package. It uses `\rgltxdoc@DescRec` and `\rgltxdoc@DescRec@i` (both with the same argument lists) for the parsing of arguments and for recursively grabbing the arguments for the `<n>` entities. At first, `\rgltxdoc@Desc` creates some vertical space above a list of description headers. Afterwards it starts the recursion.

```

93 \RequirePackage{pbox}
94 \newcommand\rgltxdoc@Desc{\medskip\par\noindent\rgltxdoc@DescRec}
95 \newcommand\rgltxdoc@DescRec[4]{%
96   \@ifnextchar[%]
97     {\rgltxdoc@DescRec@i{#1}{#2}{#3}{#4}}%
98     {\rgltxdoc@DescRec@i{#1}{#2}{#3}{#4}[]}}
99 \def\rgltxdoc@DescRec@i#1#2#3#4[#5]#6#7{%

```

The following code creates the “margin” text (more precisely, a box to the left of the text) and the *args* next to it.

```
100 \rgltxdoc@inmargin{\ttfamily #1{#6}}%
```

If there is no *args*, then the margin part is moved towards the left by a \quad.

```
101 {\ifstrempy{#7}{\quad}{}}%
```

```
102 #2{#7}\relax
```

Next, the index entries are created, through the comma-separated *idx* if this optional argument is given.

```
103 \ifstrempy{#5}%
```

```
104 {#3{#6}}%
```

```
105 {\forcsvlist{#3}{#5}}%
```

Next, we check whether $\langle n \rangle > 1$ and recurse, after a line break, if this is satisfied.

```
106 \ifnumgreater{#4}{1}%
```

```
107 {\\\rgltxdoc@DescRec{#1}{#2}{#3}{#4-1}}%
```

Finally, the following code ends a list of description headers, taking into account that an empty *args* allows the documentation text to already start in the same line as the “margin” text.

```
108 {\ifstrempy{#7}{\smallskip\par\noindent}\ignorespaces}}
```

`\rgltxdoc@inmargin` The `\rgltxdoc@inmargin{<text>}{<spacing>}` macro puts *text* into the margin of a newly started paragraph and uses *spacing* to put additional horizontal spacing between *text* and the left side of the paragraph.

```
109 \newcommand\rgltxdoc@inmargin[2]{%
```

```
110 \leavevmode\ull\hbox to\z@{\hss%
```

```
111 \pbox[t]{3\marginparwidth}{#1}%
```

```
112 #2}}
```

5.2 Arguments, Keys, and Values

Longer descriptions of macro/environment arguments as well as of keys (in key-value lists) and special values can be typeset in tables. For a common appearance, the `keyvaltable` package is used.

```
\begin{KeyValTable}{KeyDesc}
```

```
\end{KeyValTable}
```

This table is used for describing keys in key-value lists. It has three columns: key, desc, and default. The former two have the obvious meaning. The latter allows for specifying a default value for the key that is used when the key is not provided.

```
\begin{KeyValTable}{ValDesc}
```

```
\end{KeyValTable}
```

This table is used for describing special values (constants). It has two columns, val and desc, with their obvious meaning.

```
\begin{KeyValTable}{ArgDesc}
```

```
\end{KeyValTable}
```

This table is used for describing arguments of macros and environments in a structured fashion. It has two columns, arg and desc. Examples of this kind of table can be found in [Section 5.1](#).

Implementation The `keyvaltable` package is used for creating the tables that document keys, values etc.

```
113 \RequirePackage{keyvaltable}
114 \kvtSet{headbg=black!10,rowbg=white!5}
```

The following code defines the table types. The code should be self-explanatory in terms of which columns exist and what their alignment and purpose is.

```
115 \NewKeyValTable{KeyDesc}{%
116   key: align=l,    format=\texttt, head=\textbf{Key};
117   desc: align=X,   head=\textbf{Description and Possible Values};
118   default: align=l, format=\texttt, head=\textbf{Default};
119 }
120 \NewKeyValTable[showhead=false]{ValDesc}{%
121   val: align=l,    format=\texttt, head=\textbf{Value};
122   desc: align=X,   head=\textbf{Description};
123 }
124 \NewKeyValTable[showhead=false]{ArgDesc}{%
125   arg: align=l,   head=\textbf{Argument};
126   desc: align=X,  head=\textbf{Description};
127 }
```

5.3 Individual Entities

`\env` The `\env{environment}` macro is the counterpart of `\cs` for environment names instead of command names.

```
128 \newcommand\env[1]{\texttt{#1}}
```

`\pkgname` The `\pkgname{package-name}` macro typesets package names in a uniform font (sans-serif). Moreover, the package checks whether the package actually exists, in order to identify embarrassing typos in the package name.

```
129 \newrobustcmd\pkgname[1]{%
130   \IfFileExists{#1.sty}
131   {\textsf{#1}}
132   {\rgltxdoc@err{Package `#1' not found. Spelling?}}}
```

`\pkgnames` The `\pkgnames{package-names}` macro typesets a comma-separated list of package names.

```
133 \newcommand\pkgnames{%
134   \def\do##1{\pkgname{##1}}\def\do####1{, \pkgname{####1}}}%
135   \docsvlist}
```

`\cmarg` The `\cmarg{const-arg}` and `\coarg{const-arg}` macros are counterparts for `\marg` and `\oarg`. They format constant argument values, though.

```
136 \newcommand\cmarg[1]{\mbox{\texttt{\string{#1}\string{}}}}
137 \newcommand\coarg[1]{\mbox{\texttt{[#1]}}}
```

The following enables references to various L^AT_EX tools in the common formatting of their names.

```
138 \RequirePackage{hologo}
```


6 Typesetting Examples

For typesetting examples, the `showexpl` package is used. Some specific settings for the appearance of the example listings are defined and some auxiliary macros simplify special examples.

Generally, code examples shall be typeset in one of two ways:

1. through `lstlisting` environments, if only code shall be displayed but no visualization of the code's output;
2. through `LTXexample` environments, if the code as well as its output shall be displayed.

Below follows an example of `LTXexample` that uses some of the features provided by `rgltxdoc` on top of `showexpl`: Labels/references and sections.

```
\section{Test}
\label{sec:test}
\cref{sec:test} has number~\ref{sec:test}.
```

1 Test

Section 1 has number 1.

The following code performs the setup for both (because `LTXexample` builds on `lstlisting`).

```
139 \RequirePackage{showexpl}
140 \lstset{%
141   gobble=2,
142   frame=trbl,
143   backgroundcolor=\color{black!5!white},
144   explpreset={%
145     numbers=none, columns=flexible, basicstyle=\footnotesize\ttfamily},
146   numbers=none, columns=flexible, basicstyle=\footnotesize\ttfamily,
147   preset={\rgltxdoc@ExampleFix\rgltxdoc@SaveSecs\small\sffamily},
148   overhang=2cm,
149   pos=r,
150   captionpos=b}
```

The following enables references to `LTXexample` and `lstlisting` environments through `\cref` and `\vref`.

```
151 \crefname{lstlisting}{Listing}{Listings}
```

The following adds the `morepreset` key to listing environments, to allow for extending preset code rather than overwriting it.

```
152 \lst@Key{morepreset}\relax{\appto\SX@preset{#1}}
```

`\rgltxdoc@ExampleFix` The `\rgltxdoc@ExampleFix` macro performs some setup to enable, to some extent, functionality that `showexpl` disables or does not implement. Concretely,

- the macro simulates labels and references, as long as labels are only referenced after they have been defined (in `LTXexample` environments, the normal label and ref mechanism is otherwise disabled);
- the macro re-enables the default `\marginpar` macro, which is disabled by `LTXexample` presumably due to its suboptimal appearance; for the cases in which the appearance can be justified, the macro is enabled.

```
153 \newcommand\rgltxdoc@ExampleFix{%
```

The fake `\label[⟨type⟩]{⟨label⟩}` macro takes the optional `⟨type⟩`, as the `cleveref` package defines it. The macro first saves the current label value in a global macro for basic `\refs`. Then, for `\crefs`, the macro also stores the label's type, either from `⟨type⟩` or from `cleveref`'s routines.

```
154 \renewcommand\label[2][]{%
155   \global\csletcs{rgltxdoc@lbl@##2}{@currentlabel}%
156   \ifstrempy{##1}
157     {\csxdef{rgltxdoc@lbltype@##2}{\rgltxdoc@curlbltype}}
158     {\csgdef{rgltxdoc@lbltype@##2}{##1}}}%
```

The `\ref` and `\cref` macros simply use the values stored by `\label`. Note that the multitude of further `cleveref` and `varioref` macros, e.g., `\crefrange` are currently not implemented. They would need to be defined when there is actual demand for them.

```
159 \def\ref##1{\csuse{rgltxdoc@lbl@##1}}%
160 \def\cref##1{%
161   \csuse{cref@\csuse{rgltxdoc@lbltype@##1}@name}~\ref{##1}}%
162 \let\marginpar=\rgltxdoc@marginpar
163 }
164 \let\rgltxdoc@marginpar=\marginpar
```

The `\rgltxdoc@curlbltype` and `\rgltxdoc@curlbltype@i` macros are auxiliary macros for parsing the content of `\cref@currentlabel`, as set by the `cleveref` package. The first bracketed value in the content is the label type we're interested in here. If there's no current label type, we silently use the empty string.

```
165 \def\rgltxdoc@curlbltype{%
166   \@ifundefined{cref@currentlabel}{}
167   {\expandafter\rgltxdoc@curlbltype@i\cref@currentlabel\@nil}}
168 \def\rgltxdoc@curlbltype@i[#1][#2][#3]#4\@nil{#1}
```

```
\rgltxdoc@SaveSecs
\rgltxdoc@RestoreSecs
```

The `\rgltxdoc@SaveSecs` macro saves the section counters and the `\rgltxdoc@RestoreSecs` macro restores the values of the section counters. This allows one to use sectioning commands in code examples without interfering with the section numbering in the documentation. The `\rgltxdoc@SaveSecs` macro additionally disables the TOC macro, such that example sections do not appear in the documentation's TOC.

```
169 \newcommand\rgltxdoc@SaveSecs{%
170   \def\addcontentsline##1##2##3{%
171     \@for\SC:=chapter,section,subsection,subsubsection\do{%
172       \@ifundefined{c@\SC}{}
173       {\csedef{rgltx@ctr@\SC}{\the\value{\SC}}%
174        \setcounter{\SC}{0}}}}
175 \newcommand\rgltxdoc@RestoreSecs{%
176   \@for\SC:=chapter,section,subsection,subsubsection\do{%
177     \@ifundefined{c@\SC}{}
178     {\setcounter{\SC}{\csuse{rgltx@ctr@\SC}}}}
179 \patchcmd{\SX@resultInput}{\par}{\rgltxdoc@RestoreSecs\par}
180 {}
181 {\rgltxdoc@warn{Could not patch showexpl to reset section counters.}}
```

7 Shared Internal Code

`\rgltxdoc@err` The `\rgltxdoc@err{<error>}` macro raises the given `<error>`. The `\rgltxdoc@warn{<warning>}` macro raises the given `<warning>`.

```

182 \newcommand\rgltxdoc@err[1]{%
183   \PackageError{rgltxdoc}{#1}{}%
184 \newcommand\rgltxdoc@warn[1]{%
185   \PackageWarning{rgltxdoc}{#1}{}%

```

8 Future Work

- Add keys, as listed in KeyDesc tables to the index automatically.

Change History

v1	General: Initial version	1	\SpecialOtherIndex: enabled	
v1.1	General: load babel only if not yet loaded	1	hypdoc support	6
			\pkgnames: macro added	8
			\rgltxdoc@RestoreSecs: disabled	
			contents lines	10
v1.2	\NiceDescribeKeys: macros added	5		

Index

Symbols			\box	62
\,	53			
\@bsphack	78			
\@empty	57, 66			
\@esphack	87			
\@for	171, 176			
\@ifnextchar	96			
\@ifpackageloaded	9			
\@ifundefined	166, 172, 177			
\@nil	167, 168			
\\	48, 107			
A				
\actualchar	89, 91			
\addcontentsline	170			
\appto	152			
B				
\begingroup	54, 79			
\bgroup	24			
C				
\c@HD@hypercount	83			
\cmarg	48, 49, 136			
\cmdrgltxdoc@DescOpt@def	58			
\cmdrgltxdoc@DescOpt@init	59			
\cmdrgltxdoc@DescOpt@vals	61, 66, 68			
\coarg	136			
\color	143			
\cref	160			
\cref@currentlabel	167			
\crefname	151			
\cs	48, 49			
\csdef	44, 45			
\csedef	173			
\csgdef	158			
\csletcs	155			
\csname	37, 38, 39			

\csuse 41, 42, 159, 161, 178
\csxdef 157

D

\DeclareFontShape 25
\define@cmdkey 71, 72
\define@key 65
\do 67, 134, 171, 176
\docsvlist 70, 135

E

\egroup 24
\encapchar 81, 82, 90, 92
\endcsname 37, 38, 39
\endgroup 62, 86
\env 128
environments:
 KeyValTable 7, 7, 7
\expandafter 37, 39, 167

F

\footnotesize 145, 146
\forcsvlist 105

G

\global 155

H

\hbox 24, 57, 110
\HD@target 80
\HDorg@encapchar 81, 83
\hss 110

I

\ifbool 3, 14
\ifdef 73
\IfFileExists 130
\ifnumgreater 106
\ifstrempy .. 101, 103, 108, 156
\ignorespaces 108
\index 89, 91

K

KeyValTable (environment) 7, 7, 7
\kvSet 114

L

\label 154
\leavevmode 110

\let 57, 66, 81, 162, 164
\levelchar 91
\linewidth 60
\lst@Key 152
\lstset 140

M

\MakeOuterQuote 29
\marginpar 162, 164
\marginparwidth 111
\mbox 136, 137
\medskip 94
\meta 55

N

\NewKeyValTable .. 115, 120, 124
\NewNiceDescription 4, 36, 46, 47,
 51, 53
\newrobustcmd 129
\NiceDescribeCounter 4, 51
\NiceDescribeCounters ... 4, 51
\NiceDescribeEnv 4, 47
\NiceDescribeEnvs 4, 47
\NiceDescribeKey 4, 53
\NiceDescribeKeys 4, 53
\NiceDescribeMacro 3, 46
\NiceDescribeMacros 3, 46
\noindent 94, 108
\normalfont 55
\null 53, 110

P

\PackageError 183
\PackageWarning 185
\par 94, 108, 179
\parbox 60
\patchcmd 179
\pbox 111
\pkgname 129, 134
\pkgnames 133
\protect 89, 92
\protected@eappto 68

Q

\qqquad 57
\quad 101

R

\raggedright 61

<code>\ref</code>	159, 161	<code>\sffamily</code>	147
<code>\relax</code>	102, 152	<code>\small</code>	147
<code>\renewcommand</code>	154	<code>\smallskip</code>	108
<code>\RequirePackage</code>	1, 2, 4, 5, 8, 10, 12, 13, 15, 22, 23, 27, 28, 30, 31, 32, 33, 34, 35, 93, 113, 138, 139	<code>\SpecialEnvIndex</code>	50
<code>\rgltxdoc@marginpar</code> ..	162, 164	<code>\SpecialOtherIndex</code> ..	52, 63, 77
<code>\rgltxdoc@@sep</code>	57, 66, 68, 69, 74, 75	<code>\SpecialUsageIndex</code>	46
<code>\rgltxdoc@curlbltype</code> ..	157, 165	<code>\string</code>	46, 136
<code>\rgltxdoc@curlbltype@i</code>	167, 168	<code>\SX@preset</code>	152
<code>\rgltxdoc@Desc</code>	40, 94	<code>\SX@resultInput</code>	179
<code>\rgltxdoc@DescRec</code>	93		
<code>\rgltxdoc@DescRec@i</code> ..	97, 98, 99	T	
<code>\rgltxdoc@err</code>	132, 182	<code>\textbf</code> ..	116, 117, 118, 121, 122, 125, 126
<code>\rgltxdoc@ExampleFix</code> ..	147, 153	<code>\textcolor</code>	48, 49
<code>\rgltxdoc@inmargin</code> ...	100, 109	<code>\textrm</code>	53
<code>\rgltxdoc@nohyp@SpecialOtherIndex</code>	85, 88	<code>\textsf</code>	131
<code>\rgltxdoc@opt</code>	58, 59, 73	<code>\textsl</code>	74
<code>\rgltxdoc@RestoreSecs</code>	169	<code>\texttt</code> ..	68, 74, 116, 118, 121, 128, 136, 137
<code>\rgltxdoc@SaveSecs</code> ...	147, 169	<code>\the</code>	83, 173
<code>\rgltxdoc@warn</code>	181, 182	<code>\ttfamily</code> ..	24, 89, 92, 100, 145, 146
S		U	
<code>\SC</code> ..	171, 172, 173, 174, 176, 177, 178	<code>\usepackage</code>	64
<code>\setbox</code>	24, 57		
<code>\setcounter</code>	174, 178	V	
<code>\setkeys</code>	56	<code>\value</code>	173
<code>\setmainfont</code>	16	<code>\vmeta</code>	55
<code>\setmainlanguage</code>	6		
<code>\setmonofont</code>	18	W	
<code>\setsansfont</code>	17	<code>\wd</code>	60
		Z	
		<code>\z@</code>	24, 57, 60, 62, 110