The \texttt{rgltxdoc} package*

Richard Grewe
r-g+tex@posteo.net
January 12, 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 pdf\TeX{} and with \LaTeX{}. Other processors are untested. There is no assertion as to how close the pdf\TeX{} and \LaTeX{} results are to each other.

\begin{verbatim}
1 \RequirePackage{ifluatex}
\end{verbatim}

The \texttt{etoolbox} package is used to simplify some of the package’s code.

\begin{verbatim}
2 \RequirePackage{etoolbox}
\end{verbatim}

3 Documentation Input

The documentation is expected to be written in UTF-8 and in US-english language.

\begin{verbatim}
3 \ifbool{luatex}{
4 \RequirePackage[utf8]{luainputenc}
5 \RequirePackage{polyglossia}
6 \setmainlanguage[variant=american]{english}
7 }{
8 \RequirePackage[utf8]{inputenc}
9 \RequirePackage{english}{babel}
10 }
\end{verbatim}

*This document corresponds to \texttt{rgltxdoc v1}, dated 2019/01/05. The package is available online at \url{http://www.ctan.org/pkg/rgltxdoc} and \url{https://github.com/Ri-Ga/rgltxdoc}.

1
4 General Appearance

Code in this section determines the general appearance of documentation text and is not specific to documenting \LaTeX{} 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.

\begin{verbatim}
11 \RequirePackage[a4paper,twoside=false]{geometry}
12 \RequirePackage[DIV=9,BCOR=2.25cm]{typearea}
\end{verbatim}

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.

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

With just the above code, a construct like \texttt{\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 \texttt{\hbox} with typewriter text that is not actually displayed) and the second line declares the required font shape (see \url{https://tex.stackexchange.com/questions/234003/italic-font-in-lmodern-lighttt}).

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

Finally, \texttt{microtype} is used for small font improvements.

\begin{verbatim}
26 \RequirePackage{microtype}
\end{verbatim}

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

\begin{verbatim}
27 \RequirePackage{csquotes}
28 \MakeOuterQuote{
\end{verbatim}

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.

29 \RequirePackage{enumitem}
30 \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.

31 \RequirePackage{varioref}
32 \RequirePackage{hypdoc}
33 \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.
34 \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\[
\langle \text{idx} \rangle \{\langle \text{macro} \rangle\}\{\langle \text{parameters} \rangle\}
\]
\NiceDescribeMacros\[
\langle n \rangle\\\
\langle \text{idx}_1 \rangle \{\langle \text{macro}_1 \rangle\}\{\langle \text{params}_1 \rangle\} \ldots \langle \text{idx}_n \rangle \{\langle \text{macro}_n \rangle\}\{\langle \text{params}_n \rangle\}
\]

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\[
\langle n \rangle
\]
\NiceDescribeMacros\[
\langle \text{idx}_1 \rangle \{\langle \text{macro}_1 \rangle\}\{\langle \text{params}_1 \rangle\} \ldots \langle \text{idx}_n \rangle \{\langle \text{macro}_n \rangle\}\{\langle \text{params}_n \rangle\}
\]

The arguments to the macro are described below:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\langle n \rangle</td>
<td>This argument specifies the number of macros to be described.</td>
</tr>
<tr>
<td>\langle macro \rangle, \langle macro_1 \rangle, \ldots, \langle macro_n \rangle</td>
<td>These arguments specify the macros for which a description header shall be produced.</td>
</tr>
<tr>
<td>\langle parameters \rangle, \langle params_1 \rangle, \ldots, \langle params_n \rangle</td>
<td>These arguments take the sequence that specifies all optional and mandatory arguments of the respective \langle macro \rangle. Typically, these would be sequences of \oarg and \marg instances.</td>
</tr>
<tr>
<td>\langle idx \rangle, \langle idx_1 \rangle, \ldots, \langle idx_N \rangle</td>
<td>These optional arguments specify which index entries shall be documented, if they differ from the respective \langle macro \rangle parameters. This, for instance allows \langle macro \rangle to be “\foo(<em>)” whereas the \langle idx \rangle parameter could be “\foo, \foo</em>”:</td>
</tr>
</tbody>
</table>
\NiceDescribeEnv{\langle idx \rangle \{\langle environment \rangle \{\langle parameters \rangle\}}} 
\NiceDescribeEnvs{n}{\langle idx \rangle \{\langle env \rangle \{\langle params \rangle\}}} 

These macros are the counterparts of \NiceDescribeMacro and, respectively, \NiceDescribeMacros when it comes to \TeX environments. The \langle\langle environment\rangle\rangle (resp. \langle\langle env \rangle\rangle to \langle\langle env_n \rangle\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 6.

\NiceDescribeCounte{\langle idx \rangle \{\langle counter \rangle \{\langle qualifiers \rangle\}}} 
\NiceDescribeCounters{n}{\langle idx \rangle \{\langle ctr \rangle \{\langle qual \rangle\}}} 

These macros are analogous to the above macros, but aimed for documenting \TeX counters.

\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.

| (type) | The \langle type \rangle argument is the name of the type of entities. |
| (efmt) | The \langle efmt \rangle argument is \TeX 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. |
| (afmt) | The \langle afmt \rangle argument is \TeX 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. |
| (icmd) | The \langle icmd \rangle argument is \TeX 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**

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

The \NewNiceDescription{\langle type \rangle \{\langle efmt \rangle \{\langle afmt \rangle \{\langle icmd \rangle\}\}}} macro defines the \NiceDescribe\langle type \rangle and \NiceDescribe\langle type \rangle s macros and saves the \langle efmt \rangle and \langle afmt \rangle parameters for use by \NiceDescribe\langle type \rangle s.

```latex
\newcommand{\NewNiceDescription}[4][% 
\expandafter{\newcommand}{csname NiceDescribe#1\endcsname}{% 
\csname NiceDescribe#1$s\endcsname{1}}% 
\expandafter{\newcommand}{csname NiceDescribe#1$s\endcsname}{% 
rgltxdoc@Desc 
\csuse{rgltxdoc@@efmt@#1} 
\csuse{rgltxdoc@@afmt@#1} 
\#4)% 
\csdef{rgltxdoc@@efmt@#1}##1{#2}% 
\csdef{rgltxdoc@@afmt@#1}##1{#3} 
```

4
Macro names are formatted detokenized through \string. Arguments are formatted as is. For the index, \SpecialUsageIndex of the doc package is used.

\NewNiceDescription{Macro}{\string#1}{#1}{\SpecialUsageIndex}

Environment names are formatted with a gray \begin and \end. The arguments of environments are formatted as is. For the index, \SpecialEnvIndex of the doc package is used.

\NewNiceDescription{Env}{\textcolor{gray}{\cs{begin}}\cmarg{#1}\cmarg{#1}}{#1}{\SpecialEnvIndex}

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 \SpecialOtherIndex (see its documentation below).

\NewNiceDescription{Counter}{#1}{#1}{\SpecialOtherIndex{counter}{counters}}

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.

\newcommand\SpecialOtherIndex[3]{\@bsphack
\index{#3\actualchar{\protect\ttfamily#3}(#1)\encapchar usage}%
\index{#2:\levelchar#3\actualchar{\protect\ttfamily#3}\encapchar usage}\@esphack}

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.

\RequirePackage{pbox}
\newcommand\rgltxdoc@Desc{\medskip\par\noindent\rgltxdoc@DescRec}
\newcommand\rgltxdoc@DescRec[4]{% \@ifnextchar[\]{\rgltxdoc@DescRec@i{#1}{#2}{#3}{#4}}% \{#1\}\rgltxdoc@DescRec@i{#1}{#2}{#3}{#4}[]}
\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.
\leavevmode\null\hbox to\z@{\hss% \pbox[t]{3\marginparwidth}{\ttfamily #1}\%}
\pbox[t]{\{\marginparwidth}{\ttfamily #1\#6}}%
If there is no \{args\}, then the margin part is moved towards the left by a \quad.

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

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

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.

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.

\begin{Verbatim}
\RequirePackage{keyvaltable}
\kvtSet{headbg=black!10,rowbg=white..black!5}
\end{Verbatim}

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.

\begin{Verbatim}
\NewKeyValTable{KeyDesc}{%\key: align=1, format=\texttt, head=\textbf{Key};%\}
\end{Verbatim}
5.3 Individual Entities

The `\texttt{#1}` macro is the counterpart of `\texttt{#1}` for environment names instead of command names.

The `\texttt{#1}` 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.

The `\texttt{#1}` and `\texttt{#1}` macros are counterparts for `\texttt{#1}` and `\texttt{#1}`. They format constant argument values, though.

The following enables references to various \LaTeX tools in the common formatting of their names.

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 `\LaTeX` environments, if the code as well as its output shall be displayed.

Below follows an example of `\LaTeX` that uses some of the features provided by `rgltxdoc` on top of `showexpl`: Labels/references and sections.
The following code performs the setup for both (because LTXexample builds on lstlisting).
97 \RequirePackage{showexpl}
98 \lstset{%
99   gobble=2,
100   frame=trbl,
101   backgroundcolor=\color{black!5!white},
102   explpreset={%
103     numbers=none, columns=flexible, basicstyle=\footnotesize\ttfamily},
104   numbers=none, columns=flexible, basicstyle=\footnotesize\ttfamily,
105   preset={\rgltxdoc@ExampleFix\rgltxdoc@SaveSecs\small\sffamily},
106   overhang=2cm,
107   pos=r,
108   captionpos=b}

The following enables references to LTXexample and lstlisting environments through \cref and \vref.
109 \crefname{lstlisting}{Listing}{Listings}

The following adds the morepreset key to listing environments, to allow for extending preset code rather than overwriting it.
110 \lst@Key{morepreset}{\relax{\appto{\SX@preset}{#1}}}
not implemented. They would need to be defined when there is actual demand for
them.

\def\ref##1{%\csuse{rgltxdoc@@lbl@##1}}%
\def\cref##1{%\csuse{cref@\csuse{rgltxdoc@@lbltype@##1}@name}~\ref{##1}}%
\let\marginpar=\rgltxdoc@@marginpar
\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.

\def\rgltxdoc@curlbltype{%\@ifundefined{cref@currentlabel}{}{\expandafter\rgltxdoc@curlbltype@i\cref@currentlabel\@nil}}%
\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.

\newcommand\rgltxdoc@SaveSecs{%\@for\SC:=chapter,section,subsection,subsubsection\do{%\@ifundefined{c@\SC}{}}{\csedef{rgltx@@ctr@\SC}{\the\value{\SC}}%\setcounter{\SC}{0}}}}%
\newcommand\rgltxdoc@RestoreSecs{%\@for\SC:=chapter,section,subsection,subsubsection\do{%\@ifundefined{c@\SC}{}}{\setcounter{\SC}{\csuse{rgltx@@ctr@\SC}}}}%
\patchcmd{\SX@resultInput}{\par}{\rgltxdoc@RestoreSecs\par}\{}%
\{\rgltxdoc@warn{Could not patch showexpl to reset section counters.}}%

\section{Shared Internal Code}

\rgltxdoc@err\rgltxdoc@warn

The \rgltxdoc@err\{\textit{error}\}\} macro raises the given \textit{error}. The \rgltxdoc@warn\{\textit{warning}\}\} macro raises the given \textit{warning}.

\newcommand{\rgltxdoc@err}[1]{%\PackageError{rgltxdoc}{\textit{error}}{#1}}%
\newcommand{\rgltxdoc@warn}[1]{%\PackageWarning{rgltxdoc}{\textit{warning}}{#1}}%
Change History

v1
  General: Initial version ........ 1

Index

Symbols
\@bsphack ......................... 52
\@esphack ........................ 56
\@for .......................... 128, 133
\@ifnextchar .................... 60
\@ifundefined ...... 124, 129, 134
\@nil .......................... 125, 126
\\ ................................ 47, 72
\actualchar ..................... 53, 55
\appto .................................. 110
\bgroup .................................. 23
\cmarg ......................... 47, 48, 94
\coarg .................................. 94
\color .................................. 101
\cref .................................. 118
\cref@currentlabel ........... 125
\crefname .................................. 109
\cs .................................. 47, 48
\csdef ............................ 43, 44
\csedef .................................. 130
\csgsub .................................. 116
\csletcs .................................. 113
\csname ............................... 36, 37, 38
\csuse ......................... 40, 41, 117, 119, 135
\csxdef .................................. 115
\DeclareFontShape .......................... 24
\do .................................. 128, 133
\egroup .................................. 23
\encapchar .......................... 54, 56
\endcsname ......................... 36, 37, 38
\env .............................. 89
environments:
  KeyValTable .......... 6, 6, 6
\expandafter ............. 36, 38, 125
\F .................................. 60
\footnotesize .................. 103, 104
\forcevlist .................. 70
\global .................................. 113
\hbox .................................. 23, 64
\hss .................................. 64
\ifbool .......................... 3, 13
\IfFileExists .............. 91
\ifnumgreater .......................... 71
\ifstrempty ................. 66, 68, 73, 114
\ignorespaces .................. 73
\index .................................. 53, 55
\NewcommandName .......... 6, 6, 6
\kvtSet .................................. 75
\label .................................. 112
\leavevmode .......................... 64
\let .................................. 120, 122
\levelchar ...................... 55
\lst@Key .......................... 110
\lstset .................................. 98
\MakeOuterQuote .................. 28
\marginpar .......................... 120, 122
\marginparwidth ............ 65
\mbox .................................. 94, 95
\medskip .................................. 58