Customizing \LaTeX{} lists

Michel Goossens\(^a\), Frank Mittelbach\(^\dagger\) and Alexander Samarin\(^\ddagger\)

Abstract

In this article, which is part of the third chapter of our forthcoming book ‘A \LaTeX{} Companion’, we take a look at \LaTeX{}’s list environments. First the various parameters and commands controlling the standard \LaTeX{} lists, enumerate, itemize, and description, are discussed. Then the general list environment is introduced and we tell you how to build custom layouts by varying the values of the parameters controlling this environment.

Introductory Remarks

As mentioned above, this article is part of our book ‘A \LaTeX{} Companion’, which will be published by Addison-Wesley in November. The aim of this book is to be a real complement to Lamport’s \LaTeX{} user’s guide and reference manual and to help you use this software even more productively. At the Aston \TeX{} Conference last July, an upgrade of \LaTeX{} was announced. All supplementary commands and functionality of this new version, \LaTeX{}2e, are discussed in this book. Below we give an overview of the contents of the various chapters, whose titles should convey relatively clearly the subject area addressed in each case.

- Chapter 1: Introduction
  A short introduction to the \LaTeX{} system.

- Chapter 2: The Structure of a \LaTeX{} Document
  A discussion of generic and document-oriented markup.

- Chapter 3: Basic Formatting Tools
  \LaTeX{}’s basic typesetting commands.

- Chapter 4: The Layout of the Page
  An explanation of which tools are available to globally define the visual layout of the pages of a document by using pagestyles.

- Chapter 5: Tabular Material
  A tutorial on how to assemble material into columns and rows with the extended \texttt{tabular} and \texttt{array} environments, and their multipage equivalents—\texttt{supertabular} and \texttt{longtable}.

- Chapter 6: Mastering Floats
  A general treatment of floating material.

- Chapter 7: Font Selection
  A detailed discussion of \LaTeX{}’s New Font Selection Scheme (NFSS2) presenting its various user commands. It is shown how to add new fonts, both in math and text mode.

- Chapter 8: Higher Mathematics
  A review of the \texttt{amstex} package, which adds many powerful typesetting commands in the field of mathematics.

- Chapter 9: \LaTeX{} in a Multilingual Environment
  A discussion of the problem of using \LaTeX{} in a multi-language or non-English environment. The \texttt{babel} system and other language-specific packages are described.

- Chapter 10: Portable Graphics in \LaTeX{}
  An overview of the field of device-independent graphics showing how the \texttt{epic}, \texttt{eepic} and other packages extend the possibilities of \LaTeX{}’s basic \texttt{picture} environment.

- Chapter 11: Using PostScript
  A description of how the PostScript page description language can turn \LaTeX{} into a full blown graphics utility and how, via the NFSS, a user can choose a font from amongst hundreds of font families, available as PostScript Type 1 outlines (parts of this chapter were published in the previous MAPS).

- Chapter 12: Index Generation
  A discussion of the problems associated with preparing an index. The program \texttt{MakeIndex} is described in detail.

- Chapter 13: Bibliography Generation
  An overview of how \LaTeX{}’s companion program \texttt{BibTeX} tries to solve problems related to maintaining bibliographic data bases. Bibliographic styles are discussed and the format of the \texttt{BibTeX} language is presented in detail to enable you to customize an existing style.

- Chapter 14: \LaTeX{} Style File Documentation Tools
  A description of how to document the code of \LaTeX{} packages using the \texttt{doc} package and its companion program \texttt{docstrip}.  

© Addison-Wesley 1993

\(^a\)CN Division, CERN, CH1211 Genève 23, Switzerland, <goossens@cernvm.cern.ch>

\(^\dagger\)\LaTeX{}3 Project Coordinator, Zedemweg 62, D55128 Mainz, Germany, <mittelbach@mzdmza.zdv.Uni-Mainz.de>

\(^\ddagger\)ISO, 1, rue de Varembé, CH1211 Genève 20, Switzerland, <samarin@isocs.iso.ch>
LA T E X is presently being rewritten under the coordination of one of the authors (Frank Mittelbach), Chris Rowley and Rainer Schöpf. This endeavor is called the LA T E X3 Project (see e.g., the previous MAPS, pages 95–99). A lot of the functionality described in the ‘Companion’ as extensions to basic LA T E X will be available in that system: as part of the kernel, or in one of the extension styles. To help funding, half of the royalties from our book will go directly to the LA T E X3 Project. Therefore, when buying it, you not only obtain a handy, complete and up-to-date reference to many important and useful packages available with LA T E X today, but you also actively contribute to making LA T E X more powerful and user-friendly in the future.

1 Modifying LA T E X Lists

Lists are a very general LA T E X construct and are used to build many of LA T E X’s display-like environments. LA T E X’s standard list environments: enumerate, itemize, and description are discussed in the next section, where we also show how they can be customized. The general list environment is discussed in section 2.

It is relatively easy to customize the three standard LA T E X list environments, and the three sections below will look at each of these environments in turn. Changes to the default definitions of these environments can be made globally by redefining certain list defining parameters in the document preamble, or they can be kept local.

1.1 Customizing an enumerate List

LA T E X’s numbered list environment enumerate is characterized by the commands and representation forms shown in table 1. The first row shows the names of the counter used for numbering the four possible levels of the list. The second, and third rows are the commands giving the representation of the counters and their default definition in the standard LA T E X styles. Rows four, five, and six contain the commands, the default definition, and an example for the actual enumeration string printed by the list.

A reference to a numbered list element is constructed using the \textbf{\ref{w1}} and \textbf{\ref{w2}}, etc., respectively. The last three rows in the table show the command, its default definition, and an example for the representation of references. It is important that you are careful to take into account the definitions of both the representation and reference building commands to get the references correct.

We can now create several kinds of numbered description lists simply by applying what we have just learned.

Our first example redefines the first, and second level counters to use capital Roman digits and Latin characters. The visual representation should be the value of the counter followed by a dot. The default value of table 1 is used for the reference prefix command \textbf{\p@enumi}.}

\begin{enumerate}
\item text inside list, more text inside list, \textbf{\ref{w1}}
\item text inside list, more text inside list, \textbf{\ref{w2}}
\item text inside list, more text inside list, \textbf{\ref{w3}}
\end{enumerate}
Table 1: Commands controlling an enumerate list environment

<table>
<thead>
<tr>
<th>counter</th>
<th>First level</th>
<th>Second level</th>
<th>Third level</th>
<th>Fourth level</th>
</tr>
</thead>
<tbody>
<tr>
<td>representation</td>
<td>t\theenumi</td>
<td>t\theenumi</td>
<td>t\theenumi</td>
<td>t\theenumi</td>
</tr>
<tr>
<td>default definition</td>
<td>\arabic{enumi}</td>
<td>\alph{enumii}</td>
<td>\roman{enumiii}</td>
<td>\Alph{enumiv}</td>
</tr>
<tr>
<td>label field</td>
<td>t\labelenumi</td>
<td>t\labelenumii</td>
<td>t\labelenumiii</td>
<td>t\labelenumiv</td>
</tr>
<tr>
<td>default form</td>
<td>t\theenumi</td>
<td>(t\theenumi)</td>
<td>\labelenumi(t\theenumi)</td>
<td>p@enumii</td>
</tr>
<tr>
<td>numbering example</td>
<td>1, 2</td>
<td>(a), (b)</td>
<td>i, iv</td>
<td>A, B</td>
</tr>
<tr>
<td>prefix</td>
<td>t\p@enumi</td>
<td>t\p@enumii</td>
<td>t\p@enumiii</td>
<td>t\p@enumiv</td>
</tr>
<tr>
<td>default definition</td>
<td>\end{enumerate}</td>
<td>\item text item one level one.</td>
<td>\item text item two level one.</td>
<td>\item text item two level two.</td>
</tr>
<tr>
<td>reference example</td>
<td>\begin{enumerate}</td>
<td>\item text item one level two.</td>
<td>\item text item three level one.</td>
<td>\item text item three level two.</td>
</tr>
</tbody>
</table>

| 1 | text inside list, more text inside list, text inside list, more text inside list; | 2 | text inside list, more text inside list, text inside list, more text inside list; | 3 | text inside list, more text inside list, text inside list, more text inside list. |

Finally, for those who do not want to get involved in customizing these commands themselves, there exists a minor style \texttt{enumerate} (by David Carlisle), which redefines the enumerate environment with an optional argument specifying the style in which the counter has to be printed. This argument can contain any one of the tokens A, a, i, or 1 for typesetting the value of the counter using (respectively) the \texttt{\Alph}, \texttt{\alph}, \texttt{\Roman}, \texttt{\roman}, or \texttt{\arabic} styles. Moreover, these letters can be surrounded by any strings involving any other \TeX expressions; however the tokens A, a, I, i, or 1 must be specified inside a \{\} group if they are not to be taken literally.

The cross-reference commands \texttt{\label}, and \texttt{\ref} can be used as with the standard \texttt{enumerate} environment. Note, however, that with this style the \texttt{\ref} command only produces the chosen representation of the counter value—not the whole label. It prints the value in the same style as \texttt{\item}, as determined by the presence of one of the tokens A, a, i, or 1 in the optional argument.

This example generates the following output:

\texttt{EX i.} text item one level one. More text item one level one
\texttt{EX ii.} text item two level one.
\texttt{example a)} text item one level two. More text item one level two
\texttt{example b)} text item two level two.

This is how list entries are referenced: `\texttt{\ref{LA}}', `\texttt{\ref{LB}}' and `\texttt{\ref{LC}}' or more fully `\texttt{EX\ref{LA}}.' and `\texttt{A-\ref{LC}}'.
1.2 Customizing an itemize List

For a simple unnumbered itemize list, the labels are defined by the commands shown in table 2.

To create a list with different labels, you can redefine the label-generating command. You can make that change local for one list, as in the example below, or you can make it global by putting the \labelitemi redefinition in the document preamble. The following simple list is a standard itemize list with a marker from the PostScript ZapfDingbats font for the first level label:

```latex
\newenvironment{MYitemize}{% 
  \renewcommand{\labelitemi}{\ding{43}}% 
}{% 
\begin{itemize} \item Text of the first item in the list. \item Text of the first sentence in the second item of the list. And the second sentence. \item This sentence in the text of the third item of the list. \end{itemize} 
\begin{MYitemize}
\item Text of the first item in the list. \item Text of the first sentence in the second item of the list. And the second sentence. \item This sentence in the text of the third item of the list.
\end{MYitemize}

And this is the result:

☞ Text of the first item in the list.
☞ Text of the first sentence in the second item of the list. And the second sentence.
☞ This sentence in the text of the third item of the list.
```

1.3 Customizing a description List

Using the description environment you can change the \descriptionlabel command that generates the label. In the following example the font for typesetting the labels is changed from bold to sans serif.

```latex
\renewcommand{\descriptionlabel}{\hspace{\labelsep}\textsf{#1}}
\begin{description}
\item[A.] text inside list, text inside list, text inside list; text inside list, text inside list, more text inside list;
\item[B.] text inside list, text inside list, text inside list, text inside list, text inside list, text inside list, more text inside list;
\item[C.] text inside list, text inside list, text inside list, text inside list, text inside list.
\end{description}

The above gives:

A. text inside list, text inside list, text inside list, more text inside list;
B. text inside list, text inside list, text inside list, more text inside list;
C. text inside list, text inside list, text inside list, text inside list, text inside list.
```

The standard \LaTeX styles set the starting point of the label box in a description environment – \labelsep to the left of the left margin of the enclosing environment, so that the \labelsep command in the example above first adds a value of \labelsep to start the label aligned with the left margin.

2 Making Your Own Lists

Lists are generated by the generic environment list:

```latex
\begin{list}{\defaultlabel}{\decls}
\item_list
\end{list}
```

The parameter \defaultlabel is the text to be used as a label when an \item command is issued without an optional argument. The parameter \decls sets up the different geometrical parameters of the list environment (see Fig. 1). That figure also shows the default values for those parameters. The parameters can all be redefined with the help of the \setlength or \addtolength commands.

Several \LaTeX environments are defined with the help of list (for example quote, quotation, center, flushleft, and flushright). Note that these environments have only one item, and the \item[] command is specified in the environment definition.

As an example, we can consider the quote environment whose definition gives it the same left and right margins. The simple variant Quote, shown below, is identical to quote apart from the double quote symbols added around the text. Note the special precautions, which must be taken to eliminate undesirable white space in front (\ignorespaces) and following (\unskip) the text.

```latex
\renewcommand{\descriptionlabel}{\hspace{\labelsep}\textsf{#1}}
\begin{description}
\item[A.] text inside list, text inside list, text inside list; text inside list, text inside list, more text inside list;
\item[B.] text inside list, text inside list, text inside list, text inside list, text inside list, text inside list, text inside list, more text inside list;
\item[C.] text inside list, text inside list, text inside list, text inside list, text inside list.
\end{description}

\begin{quote}
Some quoted text, more quoted text. Some quoted text, more quoted text.
\end{quote}

\ldots text before.
```

"Some quoted text, more quoted text. Some quoted text, more quoted text."

Text following...
Table 2: Commands controlling an itemize list

<table>
<thead>
<tr>
<th>Commands</th>
<th>First level</th>
<th>Second level</th>
<th>Third level</th>
<th>Fourth level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition</td>
<td>$\textbf{\textbullet}$</td>
<td>$\ast$</td>
<td>$\cdot$</td>
<td></td>
</tr>
<tr>
<td>Representation</td>
<td>$\textbackslash{labelitemi}$</td>
<td>$\textbackslash{labelitemii}$</td>
<td>$\textbackslash{labelitemiii}$</td>
<td>$\textbackslash{labelitemiv}$</td>
</tr>
</tbody>
</table>

\begin{tikzpicture}[node distance=2cm, thick, auto]
  	
  \node (topsep) [align=center] {\texttt{\textbackslash{topsep}} \textbackslash{\textbackslash{parskip}} [+ \textbackslash{\textbackslash{partopsep}}]};
  	
  \node (labelsep) [below of=topsep,align=center] {\texttt{\textbackslash{\textbackslash{itemsep}} \textbackslash{\textbackslash{\textbackslash{\textbackslash{parsep}}}}}};
  	
  \node (item1) [below of=labelsep,align=center] {Item 1};
  \node (labelwidth) [below of=item1,align=center] {\texttt{\textbackslash{\textbackslash{\textbackslash{labelwidth}}}}};
  \node (leftmargin) [below of=labelwidth,align=center] {\texttt{\textbackslash{\textbackslash{\textbackslash{leftmargin}}}}};
  \node (rightmargin) [below of=leftmargin,align=center] {\texttt{\textbackslash{\textbackslash{\textbackslash{rightmargin}}}}};
  \node (parsep) [below of=rightmargin,align=center] {\texttt{\textbackslash{\textbackslash{\textbackslash{\textbackslash{parsep}}}}}};
  \node (listparindent) [below of=parsep,align=center] {\texttt{\textbackslash{\textbackslash{\textbackslash{\textbackslash{listparindent}}}}}};
  \node (item1label) [below of=listparindent,align=center] {Label};
  \node (item1text) [below of=item1label,align=center] {Item 1};
  \node (item1par) [below of=item1text,align=center] {Paragraph 1};
  \node (item2label) [below of=item1label,align=center] {Label};
  \node (item2text) [below of=item2label,align=center] {Item 2};
  \node (item2par) [below of=item2text,align=center] {Paragraph 2};

\end{tikzpicture}

\begin{itemize}
  \item Text preceding list
  \item \texttt{\textbackslash{\textbackslash{itemsep}} \textbackslash{\textbackslash{\textbackslash{\textbackslash{\textbackslash{parsep}}}}}}
  \item Text following list
\end{itemize}

Figure 1: The structure of a general list

**Vertical lengths**
All the vertical spaces below are rubber lengths with a value depending on the type size and the level of the list.

\texttt{\textbackslash{topsep}} Space between first item and preceding paragraph.
\texttt{\textbackslash{partopsep}} Extra space added to \texttt{\textbackslash{topsep}} when environment starts a new paragraph.
\texttt{\textbackslash{itemsep}} Space between successive items.
\texttt{\textbackslash{parsep}} Space between paragraphs within an item.

**Horizontal lengths**
\texttt{\textbackslash{leftmargin}} space between left margin of enclosing environment (or of page if top level list) and left margin of this list. Must be nonnegative. Its value depends on the list level.
\texttt{\textbackslash{rightmargin}} similar to \texttt{\textbackslash{leftmargin}} but for the right margin. Its value is usually 0pt.
\texttt{\textbackslash{listparindent}} extra indentation at beginning of every paragraph of a list except the one started by \texttt{\textbackslash{item}}. Can be negative! Its value is usually 0pt.
\texttt{\textbackslash{itemindent}} extra indentation added in front of an item label (not shown). Its value is usually 0pt.
\texttt{\textbackslash{labelwidth}} the nominal width of the box containing the label. If the natural width of the label is $\leq$ \texttt{\textbackslash{labelwidth}}, then the label is typeset flush right inside a box of width \texttt{\textbackslash{labelwidth}}. Otherwise, a box of the natural width is employed, which causes an indentation of the text on that line.
\texttt{\textbackslash{labelsep}} the space between the end of the label box and the text of the first item. Its default value is 0.5em.
General lists are often used for documenting computer commands or program functions. For instance, in the following examples entry and its variants are used. In each case the name of the topic being described is entered as the parameter of the \item command.

In the list below, the \makelabel command and the two geometrical parameters (\labelwidth and \leftmargin) are redefined.\footnote{In this and some of the following examples, we have used the style files calc and ifthen.}

\begin{verbatim}
\newcommand{\entrylabel}[1]{
\mbox{\textsf{#1:}}\hfil}
\newenvironment{entry}{\begin{list}{}{% 
\renewcommand{\makelabel}{\entrylabel}
\setlength{\labelwidth}{35pt}
\setlength{\leftmargin}{\labelwidth+\labelsep}}}{\end{list}}
\begin{entry}
\item[Description] Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.
\item[Errors] None.
\item[Return values] Any arguments in effect are passed back to the caller.
\end{entry}
\end{verbatim}

Description: Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.

Errors: None.

Return values: Any arguments in effect are passed back to the caller.

This example shows a typical problem with description-like lists when the text in the label (term) is wider than the width of the label. Standard \LaTeX lets the text of the term continue into the text of the description part. This is normally not desired, and to improve the visual appearance of the list we have started the description part on the next line. A new line was forced by putting an empty box on the same line, followed by `\par' command.

In the remaining part of this section various possibilities for controlling the width and mutual positioning of the term and description parts will be investigated. One method for accomplishing this is to change the width of the label. The environment is declared with an argument specifying the desired width of the label field (normally chosen to be the widest term entry). Note the redefinition of the \makelabel command where you specify how the label will be typeset. As this redefinition is put inside the definition of the \Ventry environment, the argument placeholder character \# must be escaped to \#\# to signal \LaTeX that you are referring to the argument of the \makelabel command, and not to the argument of the outer environment.

\begin{verbatim}
\newlength{\Mylen}
\newcommand{\Lentrylabel}[1]{\settowidth{\Mylen}{\textsf{#1:}}\ifthenelse{\dimengreater{\Mylen}{\labelwidth}}{\parbox[b]{\labelwidth}}{\makebox[0pt][l]{\textsf{#1:}}\mbox{}\par}}
\newenvironment{Lentry}{\renewcommand{\entrylabel}{\Lentrylabel}\begin{entry}}{\end{entry}}
\begin{Lentry}
\item[Description] Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.
\item[Errors] None.
\item[Return values] Any arguments in effect are passed back to the caller.
\end{Lentry}
\end{verbatim}

\begin{verbatim}
\newenvironment{Ventry}[1]{\begin{list}{}{% 
\renewcommand{\makelabel}{\entrylabel}
\setlength{\labelwidth}{\textwidth-#1-\leftmargin-\labelsep}}}{\end{list}}
\begin{Ventry}{35pt}
\item[Description] Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.
\item[Errors] None.
\item[Return values] Any arguments in effect are passed back to the caller.
\end{Ventry}
\end{verbatim}

However, several lists with varying widths for the label field on the same page might look typographically unacceptable. Evaluating the width of the term is another possibility. If it is wider than \labelwidth, an additional empty box is appended with the effect that the description part starts on a new line. This matches the conventional method for displaying options in UNIX manuals.

\begin{verbatim}
\newlength{\Mylen}
\newcommand{\Lentrylabel}[1]{\settowidth{\Mylen}{\textsf{#1:}}\ifthenelse{\dimengreater{\Mylen}{\labelwidth}}{\parbox[b]{\labelwidth}}{\makebox[0pt][l]{\textsf{#1:}}\hfil\relax}}
\newenvironment{Lentry}{\renewcommand{\entrylabel}{\Lentrylabel}\begin{entry}}{\end{entry}}
\begin{Lentry}
\item[Description] Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.
\item[Errors] None.
\item[Return values] Any arguments in effect are passed back to the caller.
\end{Lentry}
\end{verbatim}

As the last line in this example shows, the \Lentry environment is defined in terms of the \entry environment. The label generating command \entrylabel is now replaced by the \Lentrylabel command. The latter first sets the length variable \Mylen equal to the width of the label. It then compares that length
with \labelwidth. If the label is smaller than \labelwidth, then it is typeset on the same line as the description term, otherwise it is typeset in a zero width box with the material sticking out to the right as far as needed (forcing a new line) so that the description term starts one line lower.

Description:
Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.

Errors: None.
Return values: Any arguments in effect are passed back to the caller.

Yet another possibility is to allow multiline labels. We, once more, use the entry environment as a basis, but this time the command \Mentrylabel replaces the \entrylabel command. The idea here is that large labels may be split over several lines. Certain precautions have to be taken to allow hyphenation of the first word in a paragraph, and therefore the \hspace{0pt} command is introduced in the definition. The material gets typeset inside a paragraph box of the correct width \labelwidth, which is then top aligned and left adjusted into a box that is itself placed inside a box with a depth of 1 em and no height. In this way, \LaTeX does not realize that the material extends below the first line.

\newcommand{\Mentrylabel}[1]%
{\raisebox{0pt}[]\parbox[t]{\labelwidth}{%\hspace{0pt}\textsf{#1:}}}
\newenvironment{Mentry}{\renewcommand{\entrylabel}{\Mentrylabel}\begin{entry}}{\end{entry}}
\begin{Mentry}%
\item[Description] Returns from a function. If issued at top-level, the interpreter simply terminates, just as if end of input had been reached.
\item[Errors] None.
\item[Return values] Any arguments in effect are passed back to the caller.
\end{Mentry}

De-...