1 Introduction

When I began to use \LaTeX{} several year ago, two particular points annoyed me because I found customizing them was very complicated —headlines/footlines and lists. A new way to redefine the former is accomplished in my own titlesec package, but none was available to customize the latter except:

- \texttt{enumerate}, which just allows to change the label and it does it pretty well.
- \texttt{mdwlist}, which only “provides some vaguely useful list-related commands and environments,” as its manual states, and not a coherent way of handling lists.
- \texttt{paralist}, which provides lists within a paragraph (the original purpose of this package), a few other hard-wired specific changes and the optional argument of \texttt{enumerate}.

One of the main drawbacks of the standard \texttt{list} is its weird parameters, whose meaning is not always obvious. In order to provide a cleaner interface two approaches were possible: either defining new lists, or introducing a new syntax making the standard lists easier to customize. For marks I took the first approach in titlesec, just because I did not manage to find a satisfactory solution with the \LaTeX{} internal macros, but since lists are in some sense more “complete” than sections and marks, I have taken here the second approach.

In the interface a sort of “inheritance” is used. You can set globally the behaviour of lists and then override several parameters of, say, enumerate and then in turn override a few paremeters of a particular instance. The values will be searched in the hierarchy.

2 The package

This package is intended to ease customizing the three basic list environments: \texttt{enumerate}, \texttt{itemize} and \texttt{description}. It extends their syntax to allow an optional argument where a set of parameters in the form \texttt{key=value} are available:

- Vertical spacing:
  - \texttt{topsep}
  - \texttt{partopsep}
  - \texttt{parsep}
  - \texttt{itemsep}

\footnote{For bug reports, comments and suggestions go to \url{http://www.tex-tipografia.com/enumitem.html} English is not my strong point, so contact me when you find mistakes in the manual. Other packages by the same author: \texttt{gloss} (with José Luis Díaz), \texttt{accents}, \texttt{tensind}, \texttt{esindex}, \texttt{dotlessi}, \texttt{titlesec}, \texttt{titletoc}.}
For example:
\begin{itemize}[itemsep=1ex,leftmargin=1cm]
The keys above are equivalent to the well known list parameters—see a \LaTeX\ manual for a description of them. Next sections explains the extensions provided by \texttt{enumitem}.

3 Keys

This section describes the keys in displayed lists. Most of them are available in inline lists, where further keys are available (see \texttt{[\ref{enumitem}]}).

3.1 Label and cross references format

\texttt{\textbackslash label=⟨commands⟩}

Sets the label to be used in the current level. A set of starred versions of \texttt{\textbackslash alph}, \texttt{\textbackslash Alph}, \texttt{\textbackslash arabic}, \texttt{\textbackslash roman} and \texttt{\textbackslash Roman}, without argument stand for the current counter in \texttt{enumerate}.

Thus
\begin{enumerate}[label=\emph{\textbackslash alph*})]
\end{enumerate}

prints \texttt{a), b), and so on} (this is a standard style in Spanish, and formerly used by Chicago, too).

It works with \texttt{\textbackslash value}, too (provided the widest label is not to be computed or \texttt{widest*} is used, see below). A fancier example (which looks ugly, but it is intended only to illustrate what is possible; requires \texttt{color} and \texttt{pifont}):
\begin{enumerate}[label=\protect\fcolorbox{blue}{yellow}{\protect\ding{\textbackslash value*}}, start=172]
\end{enumerate}

The value of \texttt{label} is a moving argument, and fragile commands must be protected except the counters. Because of that, use of \texttt{\textbackslash value} is somewhat tricky, because \texttt{\textbackslash the} or \texttt{\textbackslash ifnum} expects an actual value, which is not the case when \texttt{label} is being processed to replace internally the \texttt{*} by the form with the counter argument. The best solution is usually encapsulating the logic inside a new “counter” with the help of \texttt{\textbackslash AddEnumerateCounter}\footnote{Which is admittedly somewhat convoluted. A better way to accomplish this is on the way.}.

If you prefer setting labels like the \texttt{enumerate} package, use “short labels” (see section \texttt{6}).

\texttt{\textbackslash label*=⟨commands⟩}

Like \texttt{label} but its value is appended to the parent label. For example, the following defines a \texttt{legal} list (1., 1.1., 1.1.1., and so on):

\begin{verbatim}
\newlist{legal}{enumerate}{10}
\setlist[legal]{label*=\texttt{arabic*}.)
\end{verbatim}

\footnote{Actually, the asterisk is currently the argument but things may change. Consider them as starred variants and follow the corresponding syntax.}
By default, label sets also the form of cross references and \the... (overriding the settings in previous hierarchical levels), but you can define a different format with this key. For example, to remove the right parenthesis:

\begin{enumerate}[label=\emph{\alph*}),ref=\emph{\alph*}]

In both label and ref, the counters can be used as usual:

\begin{enumerate}[label=\thenumi.\arabic*.

or

\begin{enumerate}[label=\arabic{enumi}.\arabic*.

(provided the current level is the second one).

Note the labels are not accumulated to form the reference. If you want, say, something like 1.a from 1) as first level and a) as second level, you must set it with ref. You may use \ref{level1}.\ref{level2} with appropriate ref settings, but as Robin Fairbairns points out in the \TeX{} FAQ

\ldots [that] would be both tedious and error-prone. What is more, it would be undesirable, since you would be constructing a visual representation which is inflexible (you could not change all the references to elements of a list at one fell swoop).

This is sensible and I recommend to follow the advice, but sometimes you might want something like:

\ldots subitem \ref{level2} of item \ref{level1} \ldots

The value of ref is a moving argument, and fragile commands must be protected except the counters.

Sets the label font. Useful when the label is changed with the optional argument of \item and in description. The last command in ⟨commands⟩ can take an argument with the item label. In description class setting are in force, so you may want begin with \normalfont. A synonymous is format.


How the label is aligned (with relation to the label box edges). Three values are possible: left, the default right and parleft (a parbox of width \labelwidth with flush left text). The parameters controlling the label spacing should be properly set, either by hand or more conveniently with the * settings (see below):

\begin{enumerate}[label=\Roman*.\, align=left, leftmargin=*]

(When the label box is supposed to have its natural width, use left.)

New align types can be defined (or the existing ones redefined) with \SetLabelAlign; the predefined values are equivalent to

\footnote{Prior to version 3.0 the left alignments was incorrectly defined and the label and the text could overlap.}
If the last thing in the definition is a skip (typically \hfill), it is removed sometimes by description. If for some reason you want to avoid this, just add \null at the end.

Although primarily intended for the alignment, this command has other uses (as in the provided \parleft). For example, with the following all labels with align=right are set as superscripts:
\SetLabelAlign{right}{\has\llap{\textsuperscript{#1}}} (A new name is also possible, of course.)

If you want the internal settings for align and font be ignored, you can override the enumitem definition of \makelabel in before:
\begin{description}[before={\renewcommand\makelabel[1]{\ref{##1}}}] (Alternatively, define a macro and use \let.)

### 3.2 Horizontal spacing of labels

\leftmargin=⟨length⟩ \labelindent

This parameter is added in enumitem for the blank space from the margin of the enclosing list/text to the left edge of the label box. This means there is a redundancy because one of the parameters depends on the others, i.e., it has to be computed from the other values, as described below. There is a new counter length \labelindent which defaults to 0 pt. The five parameters are related in the following way:

\leftmargin + \itemindent = \labelindent + \labelwidth + \labelsep

Sets which value is to be computed from the others. This is done after all keys has been read. Explicit values are not lost, and so with the following hierarchical settings:

leftmargin=2em
labelindent=1em, leftmargin=! 
labelindent=! 

leftmargin is again 2em and labelindent is the computed parameter. The default is labelindent=! , but note some keys sets another value (wide and description styles).

With align=right (the default), labelindent=! and labelwidth=! behave similarly in practice.

Like before, but labelwidth is set to the width of the current label, using the default value of 0 in \arabic*, viii in \roman*, m in \alph* and similarly in uppercase forms (these values can be changed with widest, see below). Examples are:
\begin{itemize}[label=\textbullet, leftmargin=\*]
\begin{enumerate}[label=\roman*, leftmargin=\*, widest=iii]
\begin{itemize}[label=\textbullet, leftmargin=2pc, labelindent=\*]
\begin{enumerate}[label=\arabic*, leftmargin=2\parindent, labelindent=!]
\begin{enumerate}[label=\*]
\begin{enumerate}[label=\*]
\begin{enumerate}[label=\*]

4
The most useful are \texttt{labelsep=} and \texttt{leftmargin=}.
With the former the item body begins at a fixed place (namely, \texttt{leftmargin}),
while with the latter begins at a variable place depending on the label (but always the same within a list, of course).
Most of times, what you would want is \texttt{leftmargin=}.
Unfortunately, \LaTeX{} does not define a default \texttt{labelsep} to be applied to all lists—simply the current value is used.
With \texttt{enumitem} you can set default values for every list, as described below, and so, if you want to make sure \texttt{labelsep} is under your control, all you need is something like:

\begin{verbatim}
\setlist[itemsep=.5em]
labelwidth*= and labelwidth=! are synonymous.
\end{verbatim}

\begin{verbatim}
widest=(string) widest==(integer) \texttt{NEW 3.0} widest
\end{verbatim}

To be used in conjunction with the \texttt{=} values, if desired. It overrides the default value for the widest printed counter.
Sometimes, if lists are not very long, a value of a for \texttt{\alpha} is more sensible than the default m:

\begin{verbatim}
\begin{enumerate}[leftmargin=*,widest=a] % Assume standard 2nd level
\end{verbatim}

With no value, the default is restored. With \texttt{widest}, the string is built using \texttt{(integer)} as the value of the counter (e.g., with \texttt{roman}, \texttt{widest=vii} and \texttt{widest*=8} are the same).

Since \texttt{value} does not return a string but a number, \texttt{widest} and the \texttt{=} values cannot be used with it. However, with \texttt{widest*}, being a number, it is allowed.

3.3 More on horizontal spacing

Since \texttt{parindent} is not used as such inside lists, but instead is set internally to either \texttt{itemindent} or \texttt{listparindent}, when used as the value of a parameter \texttt{enumitem} returns the global value, i. e., the value it has outside the outermost list.

The horizontal space in the left margin of the current level is distributed in the following way:

\begin{verbatim}
labelindent  labelwidth  labelsep= itemindent
leftmargin
\end{verbatim}

\begin{verbatim}
labelsep==(length) \texttt{NEW 3.0}
\end{verbatim}

Remember \texttt{labelsep} spans part of \texttt{leftmargin} and \texttt{itemindent} if the latter is not zero.
This is often somewhat confusing, so a new key is provided—\texttt{labelsep=} the value is reckoned from the left margin (it just sets \texttt{\labelsep} and then adds \texttt{\itemindent} to it, but in addition later changes to \texttt{itemindent} are taken into account):

\begin{verbatim}
labelindent  labelwidth  labelsep  itemindent
leftmargin
\end{verbatim}

\begin{verbatim}
labelindent=(length) \texttt{NEW 3.0}
\end{verbatim}

Like \texttt{labelindent}, but it is reckoned from the left margin in the current list and not from that in the enclosing list/text.

\footnote{Admittedly, these figures are not exactly the clearest possible, and I intend to improve them in a future release.}
3.4 Numbering, stopping, and resuming

\texttt{start=\langle integer\rangle}

Sets the number of the first item.

\texttt{resume}

The counter continues from the previous \texttt{enumerate}, instead of being reset to 1.

\begin{verbatim}
\begin{enumerate}
  \item First item.
  \item Second item.
\end{enumerate}
\end{verbatim}

\begin{verbatim}
\begin{enumerate}[resume]
  \item Third item
\end{enumerate}
\end{verbatim}

This is done locally. If you want global resuming, see next section on series.

\texttt{resume*}

Like \texttt{resume} but the options from the previous \texttt{enumerate} are used, too. This option must be restricted to the optional argument in a environment (this is the only place where it makes sense). It should be used sparingly—if you are using it often, then very likely you want to define a new list (see \ref{sec:Enumerate}). Further keys are allowed, and in this case the saved options are overridden by those in the current list (i.e., the position of \texttt{resume*} does not matters). For example:

\begin{verbatim}
\begin{enumerate}[resume*,start=1] % or [start=1,resume*]
\end{verbatim}

uses the keys in the previous \texttt{enumerate}, but restarts the counter. If there is a series of a certain list with \texttt{resume*}, options are taken from the list previous to the first one, except for start.

3.5 Series

\begin{verbatim}
\begin{tabular}{|c|c|}
  \hline
  \texttt{series=} & \texttt{NEW 3.0} \\
  \texttt{(series-name)} & \texttt{resume**=\langle series-name\rangle} & \texttt{resume=\langle series-name\rangle} & \texttt{NEW 3.0} \\
  \hline
\end{tabular}
\end{verbatim}

A new method (3.0) of continuing lists is by means of the key \texttt{series}, so that they behave like a unit. A list with key \texttt{series} is considered the starting list and its settings are stored \textit{globally}, so that they can be used later with \texttt{resume/resume*}. All these keys take a value with the series name (which must be different from existing keys):

- \texttt{resume=\langle series-name\rangle} just continue numbering items in the series,
- \texttt{resume**=\langle series\rangle} also applies the settings of the starting list,
- \texttt{\langle series\rangle}, i.e., the series name used as a key, is an alternative to \texttt{resume*=\langle series\rangle}.

For example:

\begin{verbatim}
\begin{enumerate}[label=\arabic**(a),leftmargin=1cm,series=lafter]
  \item A
  \item B
\end{enumerate}
\end{verbatim}

You get: 1(a) 2(a). You can continue with:
\begin{enumerate}[label=\arabic*(b),resume=lafter]
  \item A
  \item B
\end{enumerate}

You get: 3(b) 4(b). (But you can use start=1, if you like.)

Note you can add further arguments, which are executed after those saved at the starting list and therefore take precedence over them – in particular, resume* itself takes precedence over a start (e.g., start=1) in the the starting list.

Every time a series is started, several commands are defined internally, so to avoid wasting resources and use the same name for non-overlapping series.

3.6 Penalties

\begin{Verbatim}
\begin{description}
\item[beginpenalty=⟨integer⟩] \midpenalty=⟨integer⟩ \endpenalty=⟨integer⟩
\end{description}
\end{Verbatim}

Set the penalty at the beginning of a list, between items and at the end of the list, respectively. Please, refer to your \LaTeX{} or \TeX{} manual about how penalties control page breaks. Unlike other parameters, when a list starts their values are not reset to the default, thus they apply to the child lists.

\begin{Verbatim}
\begin{description}
\item[before=⟨code⟩] \item[before*=⟨code⟩]
\end{description}
\end{Verbatim}

Execute code before the list starts (more precisely, in the second argument of the list environment used to define them). The unstarred form sets the code to be executed, overriding any previous value, while the starred one adds the code to the existing one (in the setting hierarchy, see below; not with relation to the enclosing list/text). It can contain, say, rules and text, but this has not been extensively tested. All calculations have been finished, and you can access and manipulate the list parameters. For example, to have both margins (left and right) set to the widest label:

\begin{Verbatim}
\setlength{\textwidth}{\textwidth - 2\leftmargin - \labelsep}
\end{Verbatim}

\begin{Verbatim}
\begin{description}
\item[after=⟨code⟩] \item[after*=⟨code⟩]
\end{description}
\end{Verbatim}

Same, but just before the list ends.

3.7 Description styles

A key available in description.

\begin{Verbatim}
\begin{description}
\item[style=⟨name⟩]
\end{description}
\end{Verbatim}

Sets the description style. ⟨name⟩ can be any of the following:

- standard: like description in standard classes, although with other classes it could be somewhat different. The label is boxed. Sets itemindent=!.
- unboxed: much like the standard description, but the label is not boxed to avoid uneven spacing and unbroken labels if they are long. Sets itemindent=!.
- nextline: if the label does not fit in the margin, the text continues in the next line, otherwise it is placed in a box of width \leftmargin − \labelsep, i.e., the item body never sticks into the left margin. Sets labelwidth=!.
- sameline: like nextline but if the label does not fit in the margin the text continues in the same line. Same as style=unboxed,labelwidth=!.
• multiline: the label is placed in a parbox whose width is leftmargin, with several lines if necessary. Same as style=standard, align=parleft, labelwidth=!.

Three caveats: (1) mixing boxed and unboxed labels has not a well-defined behaviour, (2) when nesting list all combinations are allowed but not all make sense, and (3) nesting nextline lists is not supported (it works, but its behaviour might change in the future, because the current one is not what one could expect).

3.8 Compact lists

\texttt{noitemsep nolistsep}

The key noitemsep kills the space between items and paragraphs (i.e., itemsep=0pt and parsep=0pt), while nolistsep kills all vertical spacing.

3.9 “Wide” lists

\texttt{wide [NEW 3.0] wide=(parindent)}

With this convenience key, the leftmargin is null and the label is part of the text—in other word, the items look like ordinary paragraphs.\footnote{Here labelsep sets the separation between the label and the first word. It is equivalent to align=left, leftmargin=0pt, labelindent=\parindent, listparindent=\parindent, labelwidth=0pt, itemindent=1!}

With \texttt{wide=(parindent)} you may set at once another value instead of \texttt{\parindent}. Of course, these keys can be overridden after \texttt{wide}, too: for example, remembering that with left-aligned labels the text is pushed if the they are wider than labelwidth, you can set labelwidth=1.5em for a minimal width, or instead of itemindent=1! you may prefer itemindent=*, which sets the minimal width to that of widest label. In level 2 you may prefer labelindent=2\parindent, and so on. You may also want to combine it with noitemsep or nolistsep.

4 Inline lists

\texttt{[NEW 3.5]}

Inline lists are “horizontal” lists set as ordinary text inside a paragraph. With this package you can create inline lists, as explained below, with \texttt{\newlist}, which have their own labels and counters. However, in most cases inline versions of standard lists, with the same labelling schema, will be enough – the package option inline does that.

\texttt{inline (package option)} \hfill \texttt{enumerate* itemize* description* (environments)}

With the package option inline, three environments for inline lists are defined: \texttt{enumerate*}, \texttt{itemize*}, and \texttt{description*}. They emulate the behaviour of \texttt{paralist} and \texttt{shortlist} in that labels and settings are shared with the displayed (i.e., “normal”) lists \texttt{enumerate}, \texttt{itemize} and \texttt{description}, respectively (however, remember resuming is based on environment names, not on list types). This applies only to those created with inline – inline lists created with \texttt{\newlist} as described below are independent and use their own labels and settings. Note as well inline is not required if you needn’t inline versions of standard lists.

\texttt{itemjoin=(string) itemjoin=(string) afterlabel=(string)}

Format is set with keys itemjoin (default is a space), and afterlabel (default is

\footnote{The key nolistsep, now deprecated, introduced a thin stretch, which was not the intended behaviour.} \footnote{fullwidth is deprecated.}
An additional key is \texttt{itemjoin*}, which, if set, is used instead of \texttt{itemjoin} before the last item. So, with

\begin{verbatim}
before={\unskip{: }, itemjoin={; }}, itemjoin*={, and }
\end{verbatim}

the following punctuation between items is used:

Blah blah: (a) one; (b) two; (c) three, and (d) four. Blah blah

\texttt{itemjoin} is ignored in vertical mode (i.e., in mode unboxed and just after a quote, a displayed list and the like).

\begin{verbatim}
\setlist[enumerate,\{levels\}]{\{format\}}
\setlist[itemize,\{levels\}]{\{format\}}
\setlist[description,\{levels\}]{\{format\}}
\setlist[\{levels\}]{\{format\}}
\end{verbatim}

Where \texttt{\{level\}} is the list level (one or more) in \texttt{list}, and the corresponding levels in \texttt{enumerate} and \texttt{itemize}. With no \texttt{\{levels\}}, the format applies to all of them. Here list does not mean any list but only the three ones handled by this package and those redefined by this package or defined with \texttt{\newlist} (see below). For example:

\begin{verbatim}
\setlist[noitemsep]
\setlist[\{1\}]{labelindent=\parindent} % < Usually a good idea
\setlist[itemize,\{\}]{leftmargin=*=}
\setlist[enumerate,\{\}]{label=$\triangleleft$}
\setlist[enumerate,\{1\}]{label=\arabic*., ref=\arabic*}
\setlist[enumerate,\{2\}]{label=\emph{\alph*}, ref=theenumi.\emph{\alph*}}
\setlist[enumerate,\{3\}]{label=\roman*}, ref=theenumi.\roman*
\setlist[description]{font=sffamily\bfseries}
\end{verbatim}

These setting are read in the following order: list, list at the current level, enumerate/itemize/description, and enumerate/itemize/description at the current level; if a key appears several times with different values, the last one, i.e., the most specific one, is applied. If we are resuming a series or a list with \texttt{resume*}, the saved keys are then applied. Finally, the optional argument (except \texttt{resume*}), if any, is applied.

\LaTeX{} provides a set of macros to change many of these parameters, but setting them with this package is more consistent and sometimes more flexible at the cost of being more “explicit” (and verbose).

The list specification can contain variables and counters, provided they are expandable, and counters are calc-savvy, so that if you load this package you can write things like:

\footnote{\texttt{\setenumerate}, \texttt{\setitemize} and \texttt{\setdescription} are deprecated.}
This allows defining lists within loops.

Currently, a way to discriminate the font size is not provided (\normalsize, \small...).

6 enumerate-like labels

With the package option shortlabels you can use an enumerate-like syntax, where \textbf{A, a, I, i} and \textbf{1} stand for \texttt{\textbackslash Alph*, \textbackslash alph*, \textbackslash Roman*, \textbackslash roman*} and \texttt{\textbackslash arabic*}. This is intended mainly as a sort of compatibility mode with the \texttt{enumerate} package, and therefore the following special rule applies: if the very first option (at any level) is not recognized as a valid key, then it will be considered a label with the \texttt{enumerate}-like syntax. For example:

\begin{enumerate}
  \item labelindent=\parindent
  ...
\end{enumerate}

Although perhaps not so useful, you can omit \texttt{label=} in the itemize environment under similar conditions, too:

\begin{itemize}
  \item
  ...
\end{itemize}

\SetEnumerateShortLabel{\textbackslash key}{\texttt{replacement!}}

With this command, you can define new keys (or redefine them), which is particularly useful for enumerate to be adapted to especific typographical rules or to extend it for non-Latin scripts. Here (replacement) contains one of the starred versions of counters. For example:

\SetEnumerateShortLabel{\textbackslash i}{\textsc{roman*}}

redefines \texttt{i} so that items using this key are numbered with small caps roman numerals. The key has to be a single letter.

7 Cloning the basic lists

\newlist{(name)}{(type)}{(max-depth)}
\renewlist{(name)}{(type)}{(max-depth)}

The three lists can be cloned so that you can define “logical” environments behaving like them. To define a new lists (or redefine a existing one), use \newlist (or \renewlist), where (type) is \texttt{enumerate, itemize} or \texttt{description}.

If \texttt{type=} is \texttt{enumerate}, a set of counters with names (name)i, (name)ii, (name)iii, (name)iv, etc. (depending on \texttt{max-depth}) is defined. Don't use an arbitrarily large number for \texttt{max-depth}, to avoid creating too many counters. Then you can use those counters in labels; e. g., if you have defined a list named \texttt{steps}, you can define a label with:

\texttt{label=\arabic{stepsii}.\arabic{stepsi}}

\setlist*[\texttt{\textbackslash names},\texttt{\textbackslash levels}]{\texttt{\textbackslash keys/values}}
\setlist[\texttt{\textbackslash names},\texttt{\textbackslash levels}]{\texttt{\textbackslash keys/values}}

After creating a list, you can (in fact you must, at least the label) set the new list with \setlist:
Names in the optional argument of \setlist say which lists applies the settings to, and numbers say the level (it is calc-savvy). Several lists and/or several levels can be given, and all combinations are set; e.g:
\setlist[enumerate, itemize, 2, 3] {...}

sets enumerate/2, enumerate/3, itemize/2 and itemize/3. No number (or 0) means “all levels” and no name means “all lists”; no optional argument means “all lists at all levels”.

The three inline lists have types enumerate*, itemize*, and description*, which are available always, even without the package option inline (which just defines three environments with these names).

The starred form \setlist* adds the settings to previous ones.

By default, \LaTeX has a limit of 5 nesting levels, but when cloning list this value may be too short, and therefore you may want to set a new value. In levels below the 5th (or the deepest defined by a class), the settings of the last are used (i.e., \@listvi).

8 More about counters

8.1 New counter representation

\AddEnumerateCounter{(LaTeX command)}{(internal command)}{(widest label)}

“Registers” a counter representation so that enumitem recognizes it. Intended mainly for non Latin scripts, but also useful in Latin scripts. For example:
\makeatletter
\def\ctext#1{%expandafter@ctext\csname c@#1\endcsname}
\def\ctext#1{%ifcase#1%or First%or Second%or Third%or Fourth%or Fifth%or Sixth%fi}
\makeatother
\AddEnumerateCounter{\ctext}{\@ctext}{Second}

A starred variant allows to give a number instead of a string as the widest label; for example, if the widest label is that corresponding to the value 2:
\AddEnumerateCounter*{\ctext}{\@ctmoredx}{2}

This variant is to be preferred if the representation is not a plain string but it is styled, e.g., with small caps. (The counter names can contain @ even if not a letter.)

8.2 Restarting enumerates

\restartlist{(list-name)}

Currently, with
\setlist[enumerate]{resume}

you can get a continuous numbering through a document. A new command has been added for restarting the counter in the middle of the document:
\restartlist{(list-name)}

It is based solely in the list name, not the list type, which means enumerate* as defined with the package option inline is not the same as enumerate, because its name is different.
9 Generic keys and values

With this command you can create your own (valueless) keys. For example:

\SetEnumitemKey{midsep}{topsep=3pt,partopsep=0pt}

Keys so defined can then be used like the others. Another example is multicolumn lists, with multicol:

\SetEnumitemKey{twocol}{
  itemsep=1\itemsep,
  parsep=1\parsep,
  before=\raggedcolumns\begin{multicols}{2},
  after=\end{multicols}}

(The settings for itemsep and parsep kill the stretch and shrink parts. Of course, you may want to define a new list.)

Note the package may introduce new keys in the future, so \SetEnumitemKey is a potential source of forward incompatibilities. However, it’s safe using a non-letter character other than hyphen or star in the key name (e.g., :name or 2_col).

This commands provides a further abstraction layer for the \langle\textit{key}\rangle=\langle\textit{value}\rangle pairs. With it you can define logical names which are translated to the actual value. For example, with:

\SetEnumitemValue{label}{numeric}{\arabic*.}
\SetEnumitemValue{leftmargin}{standard}{\parindent}

you might say:

\begin{enumerate}[label=numeric,leftmargin=standard]

So, you can left to the final design what label=numeric means.

10 Package options

Besides inline, ignoredisplayed, and shortlabels, the following option is available.

\texttt{loadonly}

With this package option the package is loaded but the three lists are not redefined. You can create your own lists, yet, or even redefine the existing ones.

11 Three patterns

Three list layouts could be considered very frequent. Let us apply the parameters above to define them. (Below are samples.)

The first pattern aligns the label with the surrounding \parindent while the item body is indented depending on the label and a fixed labelsep:

\begin{verbatim}
labelindent=\parindent,
leftmargin=*
\end{verbatim}

A fairly frequent variant is aligning the label with the surrounding text (rememeber labelindent is Opt by default):

\begin{verbatim}
leftmargin=*
\end{verbatim}
The former looks better in the first level while the latter seems preferable in subsequent ones. That can be easily set with
\setlist{leftmargin=*}
\setlist[1]{labelindent=\parindent} % Only the level 1

The second pattern aligns the item body with the surrounding \parindent. In this case:
\begin{itemize}
\item \leftmargin=\parindent
\end{itemize}

A third pattern would be to align the label with \parindent and the item body with 2\parindent:
\begin{itemize}
\item \labelindent=\parindent,
\item \leftmargin=2\parindent,
\item \itemsep=*
\end{itemize}

Again, a variant would be to align the label with the surrounding text and the itembody with \parindent:
\begin{itemize}
\item \leftmargin=\parindent,
\item \itemsep=*
\end{itemize}

Note here \parindent means the global value applied to normal paragraphs.

12 The trivlist issue

\LaTeX{} uses a simplified version of list named trivlist to set displayed material, like center, verbatim, tabbing, theorem, etc., even if conceptually they are not lists. Unfortunately, trivlist uses the current list settings, which has the odd side effect that changing the vertical spacing of lists also changes sometimes the spacing in these environments.

This package modifies trivlist so that the default settings for the current level (ie, those set by the corresponding \clo{} files) are set again. In standard \LaTeX{} that is usually redundant, but if we want to fine tune lists, not resetting the default values could be a real issue (particularly if you use the nolistsep option).

A minimal control of vertical spacing has been made possible with\footnote{8\setdisplayed is deprecated.}

- \setlist[trivlist,\langle level\rangle]{\langle keys/values\rangle}

but trivlist itself, which is not used directly very often, does not accept an optional argument. This feature is not intended as a full-fledged trivlist formatter.

If for some reason you do not want to change trivlist and preserve the original definition, you can use the package option ignoredisplayed.

13 Samples

In these samples we set \setlist\{noitemsep\}

\begin{verbatim}
En un lugar de la Mancha, de cuyo nombre no quiero acordarme,
no ha mucho tiempo que vivía\'{i}a un hidalgo de los de
\begin{enumerate}\[labelindent=\parindent,leftmargin=*\]
\item lanza en astillero,
\item adarna antigua,
\item rocío\'{i}n flaco, y
\item galgo corredor.
\end{enumerate}
Una olla de algo más vaca que carnero, salpicó\'{e}n las m\'{i}'\{a}s
noches, duelos y quebrantos los s\'{a}'\{a\}s bados...
\end{verbatim}

The rule shows labelindent.
En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía
un hidalgo de los de
1. lanza en astillero,
2. adarna antigua,
3. rocin flaco, y
4. galgo corredor.
Una olla de algo más vaca que carnero, salpíon las más noches, duelos y quebrantos los sábados...

With \begin{enumerate}[leftmargin=*] % labelindent=Opt by default.
The rule shows labelindent.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía
un hidalgo de los de
1. lanza en astillero,
2. adarna antigua,
3. rocin flaco, y
4. galgo corredor.
Una olla de algo más vaca que carnero, salpíon las más noches, duelos y quebrantos los sábados...

With \begin{enumerate}[leftmargin=\parindent].
The rule shows leftmargin.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía
un hidalgo de los de
1. lanza en astillero,
2. adarna antigua,
3. rocin flaco, y
4. galgo corredor.
Una olla de algo más vaca que carnero, salpíon las más noches, duelos y quebrantos los sábados...

With \begin{enumerate}[labelindent=\parindent, leftmargin=*, label=\Roman*. ,
widest=IV, align=left].
The rule shows labelindent.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía
un hidalgo de los de
I. lanza en astillero,
II. adarna antigua,
III. rocin flaco, y
IV. galgo corredor.
Una olla de algo más vaca que carnero, salpíon las más noches, duelos y quebrantos los sábados...

With \begin{enumerate}[label=\fbox{\arabic*}]. A reference to the first item is \[1]\n
En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía
un hidalgo de los de
1. lanza en astillero,
2. adarna antigua,
3. rocin flaco, y
4. galgo corredor.
Una olla de algo más vaca que carnero, salpíon las más noches, duelos y quebrantos los sábados...

With nested lists.
En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de

(l) lanza en astillero,
(l.vii) adarna antigua,
(m) rocín flaco, y
(m.viii) galgo corredor.

Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados...

---

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de

Lo primero que tenía el Quijote lanza en astillero,
Lo segundo adarna antigua,
Lo tercero rocín flaco, y
Y por último, lo cuarto galgo corredor.

Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados...

---

Same, but with sameline.

---

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de

Lo primero que tenía el Quijote lanza en astillero,
Lo segundo adarna antigua,
Lo tercero rocín flaco, y
Y por último, lo cuarto galgo corredor.

Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados...

---

Same, but with multiline. Note the text overlaps if the item body is too short.
En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de

- Lo primero que tenía el Quijote, lanza en astillero,
- Lo segundo, adarna antigua,
- Lo tercero, rocín flaco, y
- Y por último, lo cuarto, galgo corredor.

Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados...

14 Afterword

14.1 \LaTeX{} lists

As it is well known, \LaTeX{} predefines three lists: \texttt{enumerate}, \texttt{itemize} and \texttt{description}. This is a very frequent classification which can also be found in, say, HTML. However, there is a more general model based in three fields—namely, label, title, and body—, so that enumerate and itemize has label (numbered and unnumbered) but no title, while description has title but no label. In this model, one can have a description with entries marked with labels, as for example (of course, this simple solution is far from satisfactory):

\begin{enumerate}
\item \texttt{Lo primero que tenía el Quijote} lanza en astillero,
\item \texttt{Lo segundo}, adarna antigua,
\item \texttt{Lo tercero}, rocín flaco, y
\item \texttt{Y por último, lo cuarto}, galgo corredor.
\end{enumerate}

Esta format is not infrequent at all and a tool for defining them is on the way and at a very advanced stage. It has not been include in version 3.0 because I'm not sure if the proper place is this package or \texttt{titlesec} and it is not stable enough yet.

14.2 Known issues

- List resuming is based on environment names, and when a \texttt{\newenvironment} contains a list you may want to use \texttt{\begin} and \texttt{\end}. Using the corresponding commands, however, is not an error, but it is your responsibility to make sure the result is correct.
- It seems there is no way to catch a misspelled name in \texttt{\setlist} and a meaningless error “Missing number, treated as zero” is raised.
- The behaviour of mixed boxed labels (including enumerate and itemize) and unboxed labels is not well-defined. The same applies to boxed and unboxed inline lists (which could even raise an error). Similarly, resuming a series and a list at the same time is allowed, too, but again its behaviour is not well-defined.
- (3.5.2) An incompatibility with 2.x has popped up – if you were using the optional argument to pass a value to a \texttt{\ref} or other macro requiring expandable macros, an error is raised. A quick fix is letting \texttt{\makelabel} to \texttt{\descriptionlabel} in \texttt{before}.

14.3 What’s new in 3.0

- Inline lists, with keys to set how items are joined (ie, the punctuation between items). Two modes are provided: \texttt{boxed} and \texttt{unboxed}.  

16
• \texttt{\setlist} is calc-savvy (eg, for use in loops), and you can set different lists and levels at once.
• All lengths related to labels can take the value * (and not only \texttt{labelsep} and \texttt{leftmargin}). Its behaviour has been made consistent and there is a new value ! which does not compute the widest label.
• With \texttt{\restartlist{\textit{list-name}}}, list counters can be restarted (in case you are using \texttt{\reset}).
• \texttt{\reset*} can be combined with other keys.
• Lists can be gathered globally using series, so that they are considered a single list. To start a series just use \texttt{\series=\textit{series-name}} and then resume it with \texttt{\resume=\textit{series-name}} or \texttt{\resume*=\textit{series-name}}.
• The “experimental” \texttt{fullwidth} has been replaced by a new key \texttt{wide}.
• \SetLabelAlign defines new align values.
• You can define “abstract” values (eg, \texttt{label=numeric}) and new keys.

• (3.2) \texttt{\setlist*} is calc-savvy.
• (3.2) \texttt{\value} can be used with \texttt{\widest*}.
• (3.2) Some internal restrictions in \texttt{\arabic} and the like has been removed. It is more flexible at the cost of having a more “relaxed” error checking.

14.4 Bug fixes
• Star values (eg, \texttt{leftmargin=*}) could not be overriden and new values were ignored.
• \texttt{nolistsep} as the first of several keys was not always recognized and therefore treated like a short label (i.e., \texttt{nol\roman*stsep}).
• \texttt{labelwidth} did not always work (when there was a prior \texttt{\widest} and *)
• With \texttt{align=right} the label and the following text could overlap.
• \texttt{description} did not get the correct list level.
• At some point (2.x?) \texttt{\value*} stopped working.
• (3.1) Unfortunately, \texttt{xkeyval} “kills” \texttt{keyval}, so the lattest has been replicated in \texttt{\enumitem}.
• (3.3) Fixes a serious bug – with * neither \texttt{\itemize} nor \texttt{\description} worked.
• (3.4) Fixes bad spacing in mode boxed (misplaced \texttt{\unskip} before the first item and wrong spacefactor between items).
• (3.4) \texttt{nolistsep} did not work as intended, but since the error has been there for several years, a new key \texttt{nosep} is provided.
• (3.4) The issue with \texttt{nolistsep} with \texttt{shortlabels} (see above) was not fixed in all cases. Hopefully now it is.
• (3.5.0) Fixed the fix related to the spacefactor between items.
• (3.5.0) Fixed a problem with nested boxed inline lists.
• (3.5.1) \texttt{\reset*} only worked once, and subsequent ones behaved like \texttt{\reset}.
• (3.5.2) Fixed \texttt{\setlist*}, which didn’t work.

14.5 Acknowledgements
I wish to thank particularly the comments and suggestions from Lars Madsen, who has found some bugs, too.