

# L'extension `enumerate` \*

David Carlisle

23/07/2015

Ce fichier est maintenu par l'équipe du « L<sup>A</sup>T<sub>E</sub>X Project ». Les rapports d'anomalie peuvent être envoyés en anglais à <http://latex-project.org/bugs.html> (catégorie `tools`).

## Résumé

Cette extension ajoute un argument optionnel à l'environnement `enumerate` qui indique le style dans lequel le compteur d'énumération est mis en forme.

Une occurrence de l'un des caractères `A a I i 1` ou `1` détermine la mise en forme de la valeur du compteur avec, respectivement, `\Alph`, `\alph`, `\Roman`, `\roman` ou `\arabic`.

Ces lettres peuvent être entourées de chaînes quelconques mettant en jeu toute expression T<sub>E</sub>X. Ici, les caractères `A a I i 1` doivent être placés à l'intérieur d'un groupe `{ }` s'ils ne doivent pas avoir leur sens particulier dans ce cadre.

---

\*Ce fichier a pour numéro de version v3.00 et a été mis à jour le 23/07/2015. La première traduction, basée sur la version v3.00, a été publiée par Jean-Pierre Drucbert en 2000.

# 1 Examples

EX i.	un un un un un un un un un un un un un	<pre>\begin{enumerate}[EX i.] \item un un un un un un un       un un un un\label{LA} \item two       \begin{enumerate}[{exemple} a)] \item un sur deux un sur deux       un sur deux\label{LB} \item deux sur deux \end{enumerate} \end{enumerate}</pre>
EX ii.	deux exemple a) un sur deux un sur deux un sur deux exemple b) deux sur deux	
A-1	un	<pre>\begin{enumerate}[{A}-1] \item un\label{LC} \item deux \end{enumerate}</pre>
A-2	deux	

✖ `\label` and `\ref` may be used as with the standard `enumerate` environment. `\ref` only produces the counter value, not the whole label. `\ref` prints the value in the same style as `\item`, as determined by the presence of one of the tokens `A a I i 1` in the optional argument. In the above example `\ref{LA}`, `\ref{LB}` and `\ref{LC}` produce ‘??’, ‘??’ and ‘??’ respectively.

## 2 Macros

	<pre>1 \*package&gt;</pre>
<code>\@enlab</code>	Internal token register used to build up the label command from the optional argument.
	<pre>2 \newtoks\@enLab</pre>
<code>\@enQmark</code>	This just expands to a ‘?’. <code>\ref</code> will produce this, if no counter is printed.
	<pre>3 \def\@enQmark{?}</pre>

The next four macros build up the command that will print the item label. They each gobble one token or group from the optional argument, and add corresponding tokens to the register `\@enLab`. They each end with a call to `\@enloop`, which starts the processing of the next token.

<code>\@enLabel</code>	Add the counter to the label. #2 will be one of the ‘special’ tokens <code>A a I i 1</code> , and is thrown away. #1 will be a command like <code>\Roman</code> .
	<pre>4 \def\@enLabel#1#2{% 5   \edef\@enThe{\noexpand#1{\@enumctr}}% 6   \@enLab\expandafter{\the\@enLab\csname the\@enumctr\endcsname}% 7   \@enloop}</pre>

`\@enSpace` Add a space to the label. The tricky bit is to gobble the space token, as you can not do this with a macro argument.

```

8 \def\@enSpace{\afterassignment\@enSp@ce\let\@tempa= }
9 \def\@enSp@ce{\@enLab\expandafter{\the\@enLab\space}\@enloop}

```

`\@enGroup` Add a `{ }` group to the label.

```

10 \def\@enGroup#1{\@enLab\expandafter{\the\@enLab{#1}}\@enloop}

```

`\@enOther` Add anything else to the label

```

11 \def\@enOther#1{\@enLab\expandafter{\the\@enLab#1}\@enloop}

```

`\@enloop` The body of the main loop. Eating tokens this way instead of using `\@tfor` lets you see spaces and **all** braces. `\@tfor` would treat `a` and `{a}` as special, but not `{a}`.

```

12 \def\@enloop{\futurelet\@entemp\@enloop@}
13 \def\@enloop@{%
14   \ifx A\@entemp           \def\@tempa{\@enLabel\Alph } \else
15   \ifx a\@entemp           \def\@tempa{\@enLabel\alph } \else
16   \ifx i\@entemp           \def\@tempa{\@enLabel\roman } \else
17   \ifx I\@entemp           \def\@tempa{\@enLabel\Roman } \else
18   \ifx l\@entemp           \def\@tempa{\@enLabel\arabic} \else
19   \ifx \@sptoken\@entemp   \let\@tempa\@enSpace           \else
20   \ifx \bgroup\@entemp     \let\@tempa\@enGroup           \else
21   \ifx \@enum@\@entemp    \let\@tempa\@gobble             \else
22                           \let\@tempa\@enOther
23
24   Hook for possible extensions
25   \fi\fi\fi\fi\fi\fi\fi\fi
26   \fi\fi\fi\fi\fi\fi\fi\fi
27   \fi\fi\fi\fi\fi\fi\fi\fi
28   \fi\fi\fi\fi\fi\fi\fi\fi
29   \fi\fi\fi\fi\fi\fi\fi\fi
30   \fi\fi\fi\fi\fi\fi\fi\fi
31   \fi\fi\fi\fi\fi\fi\fi\fi
32   \fi\fi\fi\fi\fi\fi\fi\fi
33   \fi\fi\fi\fi\fi\fi\fi\fi
34   \fi\fi\fi\fi\fi\fi\fi\fi
35   \fi\fi\fi\fi\fi\fi\fi\fi
36   \fi\fi\fi\fi\fi\fi\fi\fi
37   \fi\fi\fi\fi\fi\fi\fi\fi
38   \fi\fi\fi\fi\fi\fi\fi\fi
39   \fi\fi\fi\fi\fi\fi\fi\fi
40   \fi\fi\fi\fi\fi\fi\fi\fi
41   \fi\fi\fi\fi\fi\fi\fi\fi
42   \fi\fi\fi\fi\fi\fi\fi\fi
43   \fi\fi\fi\fi\fi\fi\fi\fi
44   \fi\fi\fi\fi\fi\fi\fi\fi
45   \fi\fi\fi\fi\fi\fi\fi\fi
46   \fi\fi\fi\fi\fi\fi\fi\fi
47   \fi\fi\fi\fi\fi\fi\fi\fi
48   \fi\fi\fi\fi\fi\fi\fi\fi
49   \fi\fi\fi\fi\fi\fi\fi\fi
50   \fi\fi\fi\fi\fi\fi\fi\fi
51   \fi\fi\fi\fi\fi\fi\fi\fi
52   \fi\fi\fi\fi\fi\fi\fi\fi
53   \fi\fi\fi\fi\fi\fi\fi\fi
54   \fi\fi\fi\fi\fi\fi\fi\fi
55   \fi\fi\fi\fi\fi\fi\fi\fi
56   \fi\fi\fi\fi\fi\fi\fi\fi
57   \fi\fi\fi\fi\fi\fi\fi\fi
58   \fi\fi\fi\fi\fi\fi\fi\fi
59   \fi\fi\fi\fi\fi\fi\fi\fi
60   \fi\fi\fi\fi\fi\fi\fi\fi
61   \fi\fi\fi\fi\fi\fi\fi\fi
62   \fi\fi\fi\fi\fi\fi\fi\fi
63   \fi\fi\fi\fi\fi\fi\fi\fi
64   \fi\fi\fi\fi\fi\fi\fi\fi
65   \fi\fi\fi\fi\fi\fi\fi\fi
66   \fi\fi\fi\fi\fi\fi\fi\fi
67   \fi\fi\fi\fi\fi\fi\fi\fi
68   \fi\fi\fi\fi\fi\fi\fi\fi
69   \fi\fi\fi\fi\fi\fi\fi\fi
70   \fi\fi\fi\fi\fi\fi\fi\fi
71   \fi\fi\fi\fi\fi\fi\fi\fi
72   \fi\fi\fi\fi\fi\fi\fi\fi
73   \fi\fi\fi\fi\fi\fi\fi\fi
74   \fi\fi\fi\fi\fi\fi\fi\fi
75   \fi\fi\fi\fi\fi\fi\fi\fi
76   \fi\fi\fi\fi\fi\fi\fi\fi
77   \fi\fi\fi\fi\fi\fi\fi\fi
78   \fi\fi\fi\fi\fi\fi\fi\fi
79   \fi\fi\fi\fi\fi\fi\fi\fi
80   \fi\fi\fi\fi\fi\fi\fi\fi
81   \fi\fi\fi\fi\fi\fi\fi\fi
82   \fi\fi\fi\fi\fi\fi\fi\fi
83   \fi\fi\fi\fi\fi\fi\fi\fi
84   \fi\fi\fi\fi\fi\fi\fi\fi
85   \fi\fi\fi\fi\fi\fi\fi\fi
86   \fi\fi\fi\fi\fi\fi\fi\fi
87   \fi\fi\fi\fi\fi\fi\fi\fi
88   \fi\fi\fi\fi\fi\fi\fi\fi
89   \fi\fi\fi\fi\fi\fi\fi\fi
90   \fi\fi\fi\fi\fi\fi\fi\fi
91   \fi\fi\fi\fi\fi\fi\fi\fi
92   \fi\fi\fi\fi\fi\fi\fi\fi
93   \fi\fi\fi\fi\fi\fi\fi\fi
94   \fi\fi\fi\fi\fi\fi\fi\fi
95   \fi\fi\fi\fi\fi\fi\fi\fi
96   \fi\fi\fi\fi\fi\fi\fi\fi
97   \fi\fi\fi\fi\fi\fi\fi\fi
98   \fi\fi\fi\fi\fi\fi\fi\fi
99   \fi\fi\fi\fi\fi\fi\fi\fi
100  \fi\fi\fi\fi\fi\fi\fi\fi

```

Hook for possible extensions

```

23 \fi\fi\fi\fi\fi\fi\fi\fi
24 \fi\fi\fi\fi\fi\fi\fi\fi
25 \fi\fi\fi\fi\fi\fi\fi\fi
26 \fi\fi\fi\fi\fi\fi\fi\fi
27 \fi\fi\fi\fi\fi\fi\fi\fi
28 \fi\fi\fi\fi\fi\fi\fi\fi
29 \fi\fi\fi\fi\fi\fi\fi\fi
30 \fi\fi\fi\fi\fi\fi\fi\fi
31 \fi\fi\fi\fi\fi\fi\fi\fi
32 \fi\fi\fi\fi\fi\fi\fi\fi
33 \fi\fi\fi\fi\fi\fi\fi\fi
34 \fi\fi\fi\fi\fi\fi\fi\fi
35 \fi\fi\fi\fi\fi\fi\fi\fi
36 \fi\fi\fi\fi\fi\fi\fi\fi
37 \fi\fi\fi\fi\fi\fi\fi\fi
38 \fi\fi\fi\fi\fi\fi\fi\fi
39 \fi\fi\fi\fi\fi\fi\fi\fi
40 \fi\fi\fi\fi\fi\fi\fi\fi
41 \fi\fi\fi\fi\fi\fi\fi\fi
42 \fi\fi\fi\fi\fi\fi\fi\fi
43 \fi\fi\fi\fi\fi\fi\fi\fi
44 \fi\fi\fi\fi\fi\fi\fi\fi
45 \fi\fi\fi\fi\fi\fi\fi\fi
46 \fi\fi\fi\fi\fi\fi\fi\fi
47 \fi\fi\fi\fi\fi\fi\fi\fi
48 \fi\fi\fi\fi\fi\fi\fi\fi
49 \fi\fi\fi\fi\fi\fi\fi\fi
50 \fi\fi\fi\fi\fi\fi\fi\fi
51 \fi\fi\fi\fi\fi\fi\fi\fi
52 \fi\fi\fi\fi\fi\fi\fi\fi
53 \fi\fi\fi\fi\fi\fi\fi\fi
54 \fi\fi\fi\fi\fi\fi\fi\fi
55 \fi\fi\fi\fi\fi\fi\fi\fi
56 \fi\fi\fi\fi\fi\fi\fi\fi
57 \fi\fi\fi\fi\fi\fi\fi\fi
58 \fi\fi\fi\fi\fi\fi\fi\fi
59 \fi\fi\fi\fi\fi\fi\fi\fi
60 \fi\fi\fi\fi\fi\fi\fi\fi
61 \fi\fi\fi\fi\fi\fi\fi\fi
62 \fi\fi\fi\fi\fi\fi\fi\fi
63 \fi\fi\fi\fi\fi\fi\fi\fi
64 \fi\fi\fi\fi\fi\fi\fi\fi
65 \fi\fi\fi\fi\fi\fi\fi\fi
66 \fi\fi\fi\fi\fi\fi\fi\fi
67 \fi\fi\fi\fi\fi\fi\fi\fi
68 \fi\fi\fi\fi\fi\fi\fi\fi
69 \fi\fi\fi\fi\fi\fi\fi\fi
70 \fi\fi\fi\fi\fi\fi\fi\fi
71 \fi\fi\fi\fi\fi\fi\fi\fi
72 \fi\fi\fi\fi\fi\fi\fi\fi
73 \fi\fi\fi\fi\fi\fi\fi\fi
74 \fi\fi\fi\fi\fi\fi\fi\fi
75 \fi\fi\fi\fi\fi\fi\fi\fi
76 \fi\fi\fi\fi\fi\fi\fi\fi
77 \fi\fi\fi\fi\fi\fi\fi\fi
78 \fi\fi\fi\fi\fi\fi\fi\fi
79 \fi\fi\fi\fi\fi\fi\fi\fi
80 \fi\fi\fi\fi\fi\fi\fi\fi
81 \fi\fi\fi\fi\fi\fi\fi\fi
82 \fi\fi\fi\fi\fi\fi\fi\fi
83 \fi\fi\fi\fi\fi\fi\fi\fi
84 \fi\fi\fi\fi\fi\fi\fi\fi
85 \fi\fi\fi\fi\fi\fi\fi\fi
86 \fi\fi\fi\fi\fi\fi\fi\fi
87 \fi\fi\fi\fi\fi\fi\fi\fi
88 \fi\fi\fi\fi\fi\fi\fi\fi
89 \fi\fi\fi\fi\fi\fi\fi\fi
90 \fi\fi\fi\fi\fi\fi\fi\fi
91 \fi\fi\fi\fi\fi\fi\fi\fi
92 \fi\fi\fi\fi\fi\fi\fi\fi
93 \fi\fi\fi\fi\fi\fi\fi\fi
94 \fi\fi\fi\fi\fi\fi\fi\fi
95 \fi\fi\fi\fi\fi\fi\fi\fi
96 \fi\fi\fi\fi\fi\fi\fi\fi
97 \fi\fi\fi\fi\fi\fi\fi\fi
98 \fi\fi\fi\fi\fi\fi\fi\fi
99 \fi\fi\fi\fi\fi\fi\fi\fi
100 \fi\fi\fi\fi\fi\fi\fi\fi

```

Process the current token, then look at the next.

```

25 \@tempa}

```

`\@enhook` Hook for possible extensions. Some packages may want to extend the number of special characters that are associated with counter representations. This feature was requested to enable Russian alphabetic counting, but here I give an example of a footnote symbol counter, triggered by `*`.

To enable a new counter type based on a letter, you just need to add a new `\ifx` clause by analogy with the code above. So for example to make `*` trigger footnote symbol counting, a package should do the following.

Initialise the hook, in case the package is loaded before `enumerate`.

```

\providecommand\@enhook{}

```

Add to the hook a new `\ifx` clause that associates `*` with the `\fnsymbol` counter command.

```

\g@addto@macro\@enhook{%
  \ifx *\@entemp
    \def\@tempa{\@enLabel\fnsymbol}%
  \fi}

```

This code sequence should work whether it is loaded before or after this enumerate package. Any number of new counter types may be added in this way.

At this point we just need initialise the hook, taking care not to over write any definitions another package may already have added.

```
26 \providecommand\@enhook{}
```

`\enumerate` The new `enumerate` environment. This is the first half of the original `enumerate` environment. If there is an optional argument, call `\@@enum@` to define the label commands, otherwise call `\@enum@` which is the second half of the original definition.

```
27 \def\enumerate{%
28   \ifnum \@enumdepth >3 \@toodeep\else
29     \advance\@enumdepth \@ne
30     \edef\@enumctr{enum\romannumeral\the\@enumdepth}\fi
31   \ifnextchar[{\@@enum@}{\@enum@}]}
```

`\@@enum@` Handle the optional argument..

```
32 \def\@@enum@[#1]{%
```

Initialise the loop which will break apart the optional argument. The command to print the label is built up in `\@enlab`. `\@enthe` will be used to define `\theenumn`.

```
33   \@enlab{}\let\@enthe\@enqmark
```

The `\@enum@` below is never expanded, it is used to detect the end of the token list.

```
34   \@enloop#1\@enum@
```

Issue a warning if we did not find one of the ‘special’ tokens.

```
35   \ifx\@enthe\@enqmark\warning{The counter will not be printed.%
36     ^^J\space\@spaces\@spaces\@spaces The label is: \the\@enlab}\fi
```

Define `\labelenumn` and `\theenumn`.

```
37   \expandafter\edef\csname label\@enumctr\endcsname{\the\@enlab}%
38   \expandafter\let\csname the\@enumctr\endcsname\@enthe
```

Set the counter to 7 so that we get the width of ‘vii’ if roman numbering is in force then set `\leftmarginn` to the width of the label plus `\labelsep`.

```
39   \csname c@\@enumctr\endcsname7
40   \expandafter\settowidth
41     \csname leftmargin\romannumeral\@enumdepth\endcsname
42     {\the\@enlab\hspace{\labelsep}}%
```

Finally call `\@enum@` which is the second half of the original definition.

```
43   \@enum@}
```

`\@enum@` All the list parameters have now been defined, so call `\list`. This is taken straight from the original definition of `\enumerate`.

```
44 \def\@enum@{\list{\csname label\@enumctr\endcsname}%
45   {\usecounter{\@enumctr}\def\makelabel##1{\hss\llap{##1}}}}
```

```
46 \endpackage
```