

V1.0 2024-05-26*

©2024 by Pablo González†

CTAN: https://www.ctan.org/pkg/enumext

https://github.com/pablgonz/enumext

Abstract

This package provides "enumerated list" environments for creating "simple exercise sheets" along with "multiple choice questions", storing the \(\lambda answers \rangle \) to these in memory using the multicol package and the l3seq and l3prop modules.

4.6 Keys for minipage 4.6.1 The command \miniright . . . 1.3.1 Internal counters 4.6.2 The key miniright 1.3.2 Support for multicol The storage system 3 5.1 Keys for storage 1.3.3 Support for minipage 3 5.2 Keys for internal label and ref . . . 1.3.4 The \label and \ref system . 5.3 Keys for debugging and checking . . . 1.3.5 Support for \footnote 5.4 The command \anskey The environments provided 5.5 The environment keyans 11 2.1 The environment enumext 5.5.1 The \item* in keyans 12 5.6 The environment keyanspic 12 2.2 The environment enumext* 5.6.1 The command \anspic 13 2.3 The command \item* 5 5.7 Printing stored content 2.3.1 Keys for \item* 5 5.7.1 The command \getkeyans . . . 2.4 The command \item in enumext* . . 5 5.7.2 The command \printkeyans . 3 The command \setenumext 6 The way of non-enumerated lists 17 6 4.1 Keys for label and ref 6 Change history 19 4.2 Keys for spaces 10 Index of Documentation 20 4.2.1 Vertical spaces 7 4.2.2 Horizontal spaces 12 Index of Implementation 113

Motivation and acknowledgments

Usually it is enough to use the classic enumerate environment to generate "simple exercise sheets" or "multiple choice questions", the basic idea behind enumext is to cover three points:

- 1. To have a simple interface to be able to write "lists of exercises" with "answers".
- 2. To have a simple interface for writing "multiple choice questions".
- 3. To have a simple interface for placing "columns" and "drawings" or "tables".

This package would not be possible without Phelype Oleinik who has collaborated and adapted a large part of the code and all LTEX team for their great work and to the different members of the TeX-SX community who have provided great answers and ideas. Here a note of the main ones:

- 1. Answer given by Alan Munn in \topsep, \itemsep, \partopsep, \parsep what do they each mean (and what about the bottom)?
- 2. Answer given by Enrico Gregorio in Understanding minipages aligning at top
- 3. Answer given by Ulrich Diez in Different mechanics of hyperlink vs. hyperref
- 4. Answer given by Enrico Gregorio in Minipage and multicols, vertical alignment

^{*}This file describes a documentation for v1.0, last revised 2024-05-26.

[†]E-mail: «pablgonz@educarchile.cl».

enumext v1.0 §.1 Introduction

License and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License (lppl), version 1.3 or later (https://www.latex-project.org/lppl.txt). The software has the status "maintained".

The enumext package loads and requires multicol[3] package, need to have a modern TeX distribution such as TeX Live or MiKTeX. It has been tested with the standard classes provided by LTeX: book, report, article and letter on 10pt, 11pt and 12pt.

Introduction

In the ETeX world world there are many useful packages and classes for creating "lists of exercises", "worksheets" or "multiple choice questions", classes like exam[1] and packages like xsim[2] do the job perfectly, but they don't always fit the basic day to day needs.

In my work (and in the work of many teachers) it is common to use "simple exercise sheets" also known as "informal lists of exercises", as an example:

- 1. Factor $x^2 2x + 1$
- 2. Factor 3x + 3y + 3z
- 3. True False
 - (a) $\alpha > \delta$
 - (b) LaTeX2e is cool?
- 4. Related to Linux

- (a) You use linux?
- (b) Usually uses the package manager?
- (c) Rate the following package and class
 - i. xsim-exam
 - ii. xsim
 - iii. exsheets

Sometimes we are also interested in showing the "answers" along with the questions:

- 1. Factor $x^2 2x + 1$
- $(x-1)^2$
- 2. Factor 3x + 3y + 3z
- 3(x+y+z)
- 3. True False
 - (a) $\alpha > \delta$
 - * False (b) LaTeX2e is cool? * | Very True!
- 4. Related to Linux

- (a) You use linux?
 - Yes
- (b) Usually uses the package manager?
 - * Yes, dnf
- (c) Rate the following package and class
 - xsim-exam
 - * doesn't exist for now :(
 - xsim
 - very good
 - exsheets * obsolete

Or we are interested in referring to a specific question and its "answer", for example:

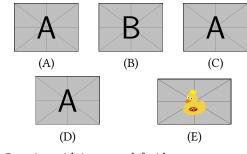
The answer to 3.(b) is "Very True!" and the answer to 4.(c).ii is "very good".

Or we are interested in printing all the "answers":

- 1. $(x-1)^2$
- 2. 3(x+y+z)
- 3. (a) False
 - (b) Very True!
- 4. (a) Yes

- (b) Yes, dnf
- (c) i. doesn't exist for now :(
 - ii. very good
 - iii. obsolete
- Another very common thing to use in my work is "multiple choice questions", for example:
- 1. First type of questions
 - (A) value
- (C) value
- (B) correct
- (D) value
- 2. Second type of questions
 - I. $2\alpha + 2\delta = 90^{\circ}$
 - II. $\alpha = \delta$
 - III. $\angle EDF = 45^{\circ}$
 - (A) I only
- (D) I and III only
- (B) II only
- (E) I, II, and III
- (C) I and II only
- ★ 3. Third type of questions
 - (1) $2\alpha + 2\delta = 90^\circ$
 - (2) $\angle EDF = 45^{\circ}$
 - (A) value
- (D) value
- (B) value (C) value
- (E) value

4. Question with image and label below:



- 5. Question with image on left side:
 - (A) value
 - (B) value
 - (C) value
 - (D) correct
 - (E) value



Where what we are interested in the $\langle label \rangle$ and a "short note" that we leave as an explanation, and then print them:

These "simple worksheets" or "multiple choice questions" appear to be easy to obtain using a combination of the enumerate, minipage and multicols environments, but like many things, what "looks simple" is not so simple.

The enumext package was created and designed to meet these small requirements in the creation of "simple worksheets" and "multiple choice questions".

1.1 Description and usage

The enumext package defines enumerated environments using the list environment provided by LTEX, but "does not redefine" any internal commands associated with it such as \list, \endlist or \item outside of the "scope" in which they are defined.

This package is NOT intend to replace the enumerate environment nor replace the powerful enumitem[5], the approach is intended to work without hindering either of them.

This package can be used with xelatex, lualatex, pdflatex and the classical latex»dvips»ps2pdf and is present in TeX Live and MiKTeX, use the package manager to install. For manual installation, download enumext.zip and unzip it, run lualatex enumext.dtx and move all files to appropriate locations, then run mktexlsr. To produce the documentation run lualatex enumext.dtx two times.

The package is loaded in the usual way:

```
\usepackage{enumext}
```

1.2 The concept of left margin

There is a direct relationship between the parameters \leftmargin, \itemindent, \labelwidth and \labelsep plus an "extra space" that makes it difficult to obtain the desired horizontal spaces in a list environment.

Usually we don't want the list to go beyond the left margin of the page, but since these four values are related, that causes a problem. The enumitem[5] package adds the \labelindent parameter to solve some of these problems. A simplified representation of this in the figure 1.



Figure 1: Representation of horizontal lengths in enumitem.

The enumext package does NOT provide a user interface to set the values for \leftmargin and \itemindent, instead it provides the keys list-offset and list-indent which internally set the values for \leftmargin and \itemindent. The concepts of \leftmargin and \itemindent are different in enumext. The figure 2 shows the visual representation of idea.



Figure 2: Representation of horizontal lengths concept in $\mbox{enumext}.$

In this way we reduce a *little* the amount of parameters we have to pass. With the default values of keys list-offset, list-indent, labelwidth and labelsep the lists will have the (usually) expected output for "simple worksheets". The figure 3 shows the visual representation.



Figure 3: Default horizontal lengths list-offset=0pt, list-indent=\labelwidth+\labelsep in enumext.

enumext v1.0 §.1 Introduction

1.3 User interface

The user interface consists in enumext, enumext*, keyans, keyans* and keyanspic environments, \anskey, \item* and \anspic* commands to $\langle stored\ content \rangle$, \getkeyans command to get the individual $\langle stored\ content \rangle$, \printkeyans to print all $\langle stored\ content \rangle$, \miniright for minipage and \setenumext to config all $[\langle key=val \rangle]$ options.

1.3.1 Internal counters

The package enumext uses internally the enumXi, enumXii, enumXii, enumXiv counters for the four nesting levels of the enumext environment, the enumXv counter for the keyans environment, the enumXvi counter for the keyanspic environment, the counter enumXvii for enumext* environment and the counter enumXviii for keyans* environment.

If any package defines these counters or they are user-defined in the document, the package will return a missing error and abort the load.

1.3.2 Support for multicol

The package provides direct support for using the multicol[3] package. This allows to obtain directly a two-column output as shown in the figure 4.



Figure 4: Representation of the two column output for a nested level in enumext environment.

The "non starred" version of the multicols environment is always used together with the \raggedcolumns command and is controlled by columns and columns-sep keys. The environment is available for all nesting levels, and can can together with the mini-env key. If you need to force a start a new column \columnbreak must be used (see §4.5).

The \columnseprule command is not available as a key and is set to "zero" for the inner levels and the keyans environment. If the value of this is set inside the document, it will affect "all environments" that use the columns key.

1.3.3 Support for minipage

The package provides direct support for minipage environment, this allows you to obtain an output like the one shown in figure 5.



Figure 5: Representation of the mini-env output for a nested level enumext environment.

The minipage environments (left and right) is always used with "aligned on top" [t], the minipage environment on the "right side" always starts with \centering. It can be used at all nesting levels and is controlled by mini-env and mini-sep keys. In order to switch from the "left" side minipage environment to the "right" side one must use the command \miniright (see §4.6).

1.3.4 The \label and \ref system

This package provides a user interface like the <code>enumitem[5]</code> package to customize the references which is activated by the <code>ref</code> key (§4.1), the standard <code>ETeX</code> \label and \ref commands work as usual. It also provides an "internal reference" system for the "stored content" by means of the key <code>save-ref</code> (§5.2) when the key <code>save-ans</code> (§5.1) is active.

The implementation of \label and \ref together with the save-ref key are compatible with the hyperref[7] package.

1.3.5 Support for \footnote

This package provides an internal implementation for the \footnote command which is compatible with the hyperref package, but, it will not produce the expected links, and when using the mini-env key or the starred environments enumext* and keyans* the output will look like the classic way they are displayed in the minipage environment.

The best way to solve this is to use Jean-François Burnol footnotehyper[8] package, it will support keeping the links if hyperref is loaded with the hyperfootnotes=true option (default) and will show the output numbered at the bottom of the page (as opposed to how it is displayed in the minipage environment). The way to load it is as follows:

```
\usepackage{footnotehyper}
\makesavenoteenv{enumext}
\makesavenoteenv{enumext*}
```

The environments provided

The package enumext provides two main list environments, the vertical environment enumext and the horizontal environment enumext*.

```
enumext*
```

```
enumext \begin{enumext} [\langle keyval \ list \rangle]
                                                                                                                  \lceil (keyval \ list) \rceil
                    \item ⟨item content⟩
                                                                                                                      \item ⟨item content⟩
                    \item [\langle custom \rangle] \langle item content \rangle
                                                                                                                      \item [\langle custom \rangle] \langle item content\rangle
                    \left\langle item^* \left[ \left\langle symbol \right\rangle \right] \left[ \left\langle offset \right\rangle \right] \right\rangle
                                                                                                                     \lceil \text{item}^* [\langle symbol \rangle] [\langle offset \rangle] \langle item content \rangle
                                                                                                                  \end{enumext}
```

The environment enumext

The enumext is an environment that works in the same way as the standard enumerate environment provided by LTFX, \item and \item[\langle custom \rangle] commands work in the usual way. The environment can be nested with at most "four levels" and the options can be configured globally using \setenumext command and locally using $\lceil \langle key = val \rangle \rceil$ in the environment.

Example with columns=2

1. This text is in the first level.

A. This text is in the fourth level.

(a) This text is in the second level.

X This text is in the first level.

This text is in the third level.

★ 2. This text is in the first level.

The environment enumext* 2.2

The enumext* environment is a horizontal list environment similar to the enumerate* environment provided by the enumitem package or task environment provided by the task package, \item and $\forall i tem[\langle custom \rangle]$ work as usual. The options can be configured globally using $\exists command$ and locally using $[\langle key = val \rangle]$ in the environment.

Some considerations to take into account for this environment:

- The environment cannot be nested within itself, but it can be nested within enumext and can contain it nested within it.
- Each "item" in the environment is placed within a minipage environment whose width is stored in the dimension \itemwidth that includes labelwith, labelsep plus the width of the content.
- You cannot have floating environments like figure or table but \footnote with hyperref support is supported if the footnotehyper package is loaded.

Example with columns=2

2. This text is in the first level. 1. This text is in the first level. X This text is in the first level. \star 3. This text is in the first level.

The command \item*

```
\item* \item*
```

```
\times [\langle symbol \rangle]
\forall item^* [\langle symbol \rangle] [\langle offset \rangle]
```

The $\forall i \neq m^* [\langle symbol \rangle]$ and $\forall i \neq m^* [\langle symbol \rangle] [\langle offset \rangle]$ works like the numbered $\forall i \neq m$, but placing a $\langle symbol \rangle$ to the "left" of the $\langle label \rangle$ separated from it by the value set by the labelsep key and can be $\langle offset \rangle$ using the second optional argument. The default values for $\langle symbol \rangle$ and $\langle offset \rangle$ are \$\star\$ '*' and the value set by labelsep key.

The starred version '*' cannot be separated by spaces 'u' from the command, i.e. \item* and the first optional argument does "not support" verbatim content. Can be configure with the keys item-sym* and item-pos* locally in the environment or globally using \setenumext command (§3).

🥑 The behavior of \item* in the enumext and enumext* environments is NOT the same as in the keyans and keyans* environments.

2.3.1 Keys for \item*

```
item-sym* = \{\langle symbol \rangle\}
```

default: \$\star\$

Sets the *symbol* to be displayed in the "left" of the box containing the current \(label \) set by labelwidth key for \item* in enumext. The symbol can be in text or math mode, for example item-sym*={\$\ast\$}.

```
item-pos* = {\langle rigid \ length \mid dim \ expression \rangle}
```

default: by levels

Sets the *offset* between the box containing the current $\langle label \rangle$ defined by labelwidth key and the $\langle symbol \rangle$ set by item-sym* key. The default values are set by labelsep key at each level. If positive values are passed it will offset to the left and if negative values are passed it will offset to the right.

The command \item in enumext*

The \item command for the enumext* environment provides an optional first argument \item($\langle number \rangle$) which "joins items" between columns. Let's consider the following examples adapted directly from the task package:

```
\begin{enumext*} [widest=10, columns=4]
  \item The first
  \item* The second
  \item The third
  \item The fourth
  \item(3)* The fifth item is way too long for this and needs three columns
  \item The sixth
  \item the seventh
  \item(2)[X] The eighth item is way too long for this and needs two columns
  \item[Z] The nineth
  \item The tenth
\end{enumext*}
```

- 1. The first
- \star 2. The second
- 3. The third
- 4. The fourth
- \star 5. The fifth item is way too long for this and needs three columns
- 6. The sixth

- X The eighth item is way too long for this and needs Z The nineth two columns
- 8. The tenth

The command \setenumext

```
\setenumext \setenumext{\langle key = val \rangle}
                                                                                                                          \star{\left(\langle keyans^* \rangle\right)} \left\{\langle key = val \rangle\right\}
                                                                                                                          \setenumext[\langle enumext, level \rangle] \{\langle key = val \rangle\}
                       \strut = \sum \{\langle enumext^* \rangle \} \{\langle key = val \rangle \}
                                                                                                                          \startion{1}{\text{\section}} \left( print, * \right) \left\{ \left\langle key = val \right\rangle \right\}
                       \strut \langle keyans \rangle ] \{ \langle key = val \rangle \}
                                                                                                                          \setenumext[\langle print^* \rangle] {\langle key = val \rangle}
```

The command \setenumext sets the $\langle keys \rangle$ on a global basis for environments enumext, enumext*, keyans, keyans* and the \printkeyans command. It can be used both in the preamble and in the body of the document as many times as desired.

The $\langle keys \rangle$ set in the optional arguments of environments and commands have the highest precedence, overriding both options passed by \setenumext. If the optional argument is not passed, the first level of the environment enumext will be taken by default.

 $m{\mathscr{G}}$ All $\langle keys
angle$ related to the "storage system" should NOT be passed through this command and should be activated directly at the "first level" of the environment in which they are executed.

The key=val system

The \(\lambda ey = val \rangle \) system used by the enumext package is implemented using \(\begin{align*} \) 3keys so it must be taken into consideration that those keys marked as "value forbidden", that is $\langle key \rangle$ is different from $\langle key \rangle$.

All \(\lambda eys \rangle \) described in this section are available for the enumext, enumext*, keyans and keyans* environments with the exception of the keys series, resume, resume* which are only available for the "first level" of the environments enumext and enumext*; and the keys mini-right, mini- right* which are only available for the enumext* and keyans* environments.

All \(\langle keys\rangle\) related to vertical or horizontal spacing accept a "skip" or "dim" expression if passed between braces, i.e. you do not need to use \dimeval or \dimexpr to perform calculations.

It should be kept in mind that using any $\langle key \rangle$ that sets a *rubber lengths* or *rigid lengths* for vertical or horizontal space on a level will influence the vertical and horizontal space for inners levels and keyans, keyans* and keyanspic environments.

4.1 Keys for label and ref

```
label = \{ \langle \text{\ } | \text{\ } |
```

default: by levels

Sets the $\langle label \rangle$ that will be printed at the *current level*. The default value for the first level of the environments enumext and enumext* are \arabic*., for second level are (\alph*), for third level are \roman*. and for fourth level are \Alph*. For keyans and keyans* environments the default value is \Alph*).

This key is intended to give the basic structure with which the $\langle label \rangle$ will be displayed, and the form in which it is used by standard "label and ref" and the "internal reference" system with the save-ref key. You cannot use commands with $\langle label \rangle$ as an argument, for example $\mbox{emph}\{\langle \mbox{alph}^* \rangle\}$ will return an error. For full customization of how $\langle label \rangle$ is displayed use the font or wrap-label keys.

```
ref = \{ \langle code \ \{ \alph^* | \arabic^* |
```

default: empty

Modifies the way *cross references* are displayed. The label key sets the default form of the *cross references*, by using this key you can define a different format, for example: $ref=\ensuremath{\texttt{ref}}\$ is valid.

Internally it renews the command associated with each counter when it is executed, i.e., in the environment enumext the command \theenumXi is modified when the key is executed at the first level, \theenumXii when it is executed at the second level and \theenumXiii together with \theenumXiv when it is executed at the third and fourth levels.

This must be kept in mind, since the values set by the label and ref keys are not cumulative by levels, so if you have used the ref key in the first level and then want to associate the counter with label or ref in the second level you must use the direct commands, i.e. \arabic{eunumXi} to indicate the count of the first level instead of using \theenumXi.

```
labelsep = \{ \langle rigid \ length \rangle \}
```

default: 0.3333em

Sets the *horizontal space* between the box containing the current $\langle label \rangle$ defined by label key and the text of an item on the first line. Internally sets the value of \labelsep for the current level.

```
labelwidth = \{\langle rigid\ length\rangle\}
```

default: by label

Sets the *width* of the box containing the current $\langle label \rangle$ set by label key. Internally sets the value of \labelwidth for the current level. The default values are calculated by means of the *width* of a box by setting a *value* to the current counter using '0' for \arabic*, 'M' for \Alph*, 'm' for \alph*, 'VIII' for \Roman* and 'viii' for \roman*.

```
widest = \{ \langle integer \mid string \rangle \}
```

default: empty

Sets the labelwidth key pass the *(integer)* or converting the *(string)* of the form *(alph, alph, alph, means* or *(roman)* to a *value* for the current counter defined by label key, then calculating the *width* by means of a box. For example widest={XXIII} or widest={23} are equivalent. This key is useful when the default values of the labelwidth key are smaller than those actually used.

```
font = \{\langle font \ commands \rangle\}
```

default: empty

Sets the *font style* for the current $\langle label \rangle$ defined by label key. For example font={\bfseries\small}.

```
align = \{ \langle left \mid right \mid center \rangle \}
```

default: left

Sets the *aligned* of $\langle label \rangle$ defined by label key on the current level in the label box.

```
wrap-label = \{ \langle code \ \{ \#1 \} \ more \ code \rangle \}
```

default: empty

Wraps the *current* $\langle label \rangle$ defined by label key referenced by $\{\#1\}$. The $\{\langle code \rangle\}$ must be passed between braces. This key does not modify the value set by the labelwidth key and is applied only on \item and \item*. When using it in the \setenumext command it is necessary to use the *double hash* ' $\{\#\#1\}$ '. For example wrap-label= $\{\footnotem\}$ or you can create a command:

```
\NewDocumentCommand \itembx { s +m }
    {%
     \IfBooleanTF{#1}
        {\strut\smash{\parbox[t]{\labelwidth}{\raggedright{#2}}}}%
        {\strut\smash{\parbox[t]{\labelwidth}{\raggedleft{#2}}}}%
}
```

and then pass it through the key wrap-label={\itembx{#1}} or wrap-label={\itembx*{#1}}.

```
wrap-label* = {\langle code \{ #1 \} \ more \ code \rangle}
```

default: empty

The same as the wrap-label key but also applies on $\idetime [\langle custom \rangle]$.

4.2 Keys for spaces

```
show-length = \{ \langle \mathit{true} \mid \mathit{false} \rangle \}
```

default: false

Displays on the terminal the values for *all list parameters* at the current level. For *vertical spaces* show the values of \topsep, \itemsep, \parsep and \partopsep. For *horizontal spaces* show the values of \labelwidth, \labelsep, \itemindent, \listparindent and \leftmargin.

4.2.1 Vertical spaces

 $topsep = \{ \langle rubber \ length \mid rigid \ length \rangle \}$

default: by levels

Set the *vertical space* added to both the top and bottom of the list. Internally sets the value of \topsep for the current level. The default value for the first level of the environments enumext and enumext* are 8.0pt plus 2.0pt minus 4.0pt, for second level are 4.0pt plus 2.0pt minus 1.0pt, for third and fourth level are 2.0pt plus 1.0pt minus 1.0pt. For keyans and keyans* environments the default value is 4.0pt plus 2.0pt minus 1.0pt.

 $parsep = \{ \langle rubber \ length \mid rigid \ length \rangle \}$

default: by levels

Set the *vertical space* between paragraphs within an item. Internally sets the value of \parsep for the current level. The default value for the first level of the environments enumext and enumext* are 4.0pt plus 2.0pt minus 1.0pt, for second level are 2.0pt plus 1.0pt minus 1.0pt, for third and fourth level are 0pt. For keyans and keyans* environments the default value is 2.0pt plus 1.0pt minus 1.0pt.

 $partopsep = \{ \langle rubber \ length \mid rigid \ length \rangle \}$

default: by levels

Set the *vertical space* added, beyond topsep, to the "top" and "bottom" of the entire environment if the environment instance is preceded by a "blank line" or \par command. Internally sets the value of \partopsep for the current level. The default values for first and second level in environment enumext are 2.0pt plus 1.0pt minus 1.0pt, for third and fourth level are 1.0pt minus 1.0pt. For keyans, keyans* and enumext* environments the default value is 2.0pt plus 1.0pt minus 1.0pt.

The value of this parameter also affects the *inner levels* and the environments keyans, keyanspic and keyans*. Caution should be taken with "blank lines" or \par command "before" each environment or nested level when formatting the source code of document. TeX will enter \(\subseteq vertical mode \rangle \) and apply this value to the "top" and "bottom" the environment or nested level.

 $itemsep = \{ \langle rubber \ length \mid rigid \ length \rangle \}$

default: by leve

Set the *vertical space* between items, beyond the parsep. Internally sets the value of \itemsep for the current level. The default value for the first level of the environments enumext and enumext* are 4.0pt plus 2.0pt minus 1.0pt, for the rest of the levels are 2.0pt plus 1.0pt minus 1.0pt. For keyans and keyans* environments the default value is 4.0pt plus 2.0pt minus 1.0pt.

noitemsep

default: not used

This is a "meta-key" that does not receive an argument. Set itemsep and parsep equal to Opt the entire level of environment.

nosep (value forbidden)

default: not used

This is a "meta-key" that does not receive an argument. Sets all keys for vertical spacing equal to opt the entire level of environment.

The following $\langle keys \rangle$ should be used with "caution", they are intended to be used at the "top" and "bottom" of the environment when the columns or mini-env keys do not provide adequate vertical spaces. The values passed can be rubber or rigid lengths, the way they are applied is the way you differ, using the star '*' $\langle keys \rangle$ applies $\langle vspace \rangle$ so that ΔT_{EX} does not discard this space at page break.

 $above = \{ \langle rubber \ length \ | \ rigid \ length \rangle \}$

default: not used

Set the *extra vertical space* added, beyond topsep, to the top of the entire level of environment. This key is intended to give a "*fine adjustment*" of the vertical space on the "*above*" the environment without hindering the value of the topsep key. The space is added with \vspace so is "*discardable*".

 $above* = \{\langle rubber\ length \mid rigid\ length\rangle\}$

default: not used

Set the *extra vertical space* added, beyond topsep, to the top of the entire level of environment. This key is intended to give a "fine adjustment" of the vertical space on the "above" the environment without hindering the value of the topsep key. The space is added with \vspace* so is "not discardable".

 $below = \{ \langle rubber\ length \mid rigid\ length \rangle \}$

default: not used

Set the *extra vertical space* space added, beyond topsep, to the bottom of the entire level of environment. This key is intended to give a "*fine adjustment*" of the vertical space on the "*below*" the environment without hindering the value of the topsep key. The space is added with \vspace so is "*discardable*".

 $below* = \{ \langle rubber\ length \mid rigid\ length \rangle \}$

default: not use

Set the *extra vertical space* space added, beyond topsep, to the bottom of the entire level of environment. This key is intended to give a "*fine adjustment*" of the vertical space on the "*below*" the environment without hindering the value of the topsep key. The space is added with \vspace* so is "not discardable".

4.2.2 Horizontal spaces

 $itemindent = \{ \langle rigid \ length \rangle \}$

default: 0pt

Extra *horizontal indentation*, beyond labelsep, of the *"first line"* off each item. This value is applied internally using \hspace and does not modify the value of \itemindent.

 $rightmargin = \{\langle rigid \ length \rangle\}$

default: 0pt

Set the *horizontal space* between the right margin of the environment and the right margin of the enclosing environment, the value it takes must be greater than or equal to opt. Internally sets the value of \rightmargin for the current level.

 $listparindent = \{ \langle \mathit{rigid} \ \mathit{length} \rangle \}$

default: 0pt

Sets the *horizontal space* indentation, beyond list-indent, for second and subsequent paragraphs within a list item. Internally sets the value of \listparindent for the current level.

 $list-offset = \{\langle rigid \ length \rangle\}$

default: 0pt

Sets the *horizontal translation* of the entire environment level from the left edge of the box defined by the labelwidth key. Internally sets the values of \leftmargin and \itemindent for the current level.

list-indent = $\{\langle rigid\ length\rangle\}$

default: labelwidth + labelsep

Sets the *indentation* of the whole environment under the box defined by labelwidth and labelsep keys. Internally sets the value of \leftmargin and \itemindent for the current level.

If list-indent=0pt the (label) will be part of the text, separated by the value of the labelsep key and the first word, in simple terms it will look like a "common paragraph". This setting is equivalent (more or less) to the wide key provided by the enumitem package.

4.3 Keys for add code

In following $\langle keys \rangle$ should be used with "caution", they are intended to inject $\{\langle code \rangle\}$ into different parts of the defined environments. We must keep in mind that the defined environments are based on the list base environment provided by ETEX which is defined (simplified) as plain form $\{\text{list}(arg\ one)\}\{\langle arg\ two\rangle\}$. Using the before* key does not allow access to the list parameters defined by $[\langle key=val\rangle]$.

before = $\{\langle code \rangle\}$

fault not used

Execute $\{\langle code \rangle\}$ "before" the environment starts. The $\{\langle code \rangle\}$ must be passed between braces, is executed "after" performing all calculations related to the *list parameters* in the environment and the parameters sets by $[\langle key=val \rangle]$ that is, in the second argument of the list after setting all the parameters $\text{list}\{\langle arg\ one \rangle\}\{\langle arg\ two \rangle\{\langle code \rangle\}\}$.

before* = $\{\langle code \rangle\}$

default: not use

Execute $\{\langle code \rangle\}$ "before" the environment starts. The $\{\langle code \rangle\}$ must be passed between braces, is executed "before" performing all calculations related to the list parameters and $[\langle key = val \rangle]$ sets in the environment that is, before the arguments defining the environment are executed: $\{\langle code \rangle\}$ \list $\{\langle arg\ one \rangle\}$ $\{\langle arg\ two \rangle\}$.

 $first = \{\langle code \rangle\}$

default: not used

Executes $\{\langle code \rangle\}$ when "starting" the environment. The $\{\langle code \rangle\}$ must be passed between braces, is executed right "after" all list parameters are done, after the second argument of list, just before the first occurrence of \item: \list{\arg one}}{\arg one}\}{\arg one}\}{\arg one}\}{\arg one}\}{\arg one}\}\item.

© Keep in mind that the code set in this key will affect the entire "body" of the environment and therefore the inner levels of the list and the keyans environment. It is recommended to set this key per level.

 $\mathsf{after} = \{ \langle \mathit{code} \rangle \}$

default: not used

Execute $\{\langle code \rangle\}$ "after" finishing the environment. The $\{\langle code \rangle\}$ must be passed between braces.

4.4 Keys for start, series and resume

 $start = \{ \langle integer \mid string \rangle \}$

default:

Sets the *start value* of the numbering on the current level. Internally $\langle string \rangle$ is passed as value to the counter defined by label key on the current level, i.e. it is equivalent to enter start=5, start=E or start=V

The following \(\lambda keys\rangle\) are "only" available for the "first level" of enumext and enumext* and are ignored if set when nested inside each other.

 $series = \{\langle series \ name \rangle\}$

default: not used

Stores the *keys* of the optional argument of the "first level" of the environment in which it is executed in $\{\langle series\ name \rangle\}$ which is used as an argument in the key resume. The $\langle keys \rangle$ stored in $\{\langle series\ name \rangle\}$ are not cumulative and are overwritten if the same $\{\langle series\ name \rangle\}$ is used again.

 $resume = \{\langle series \ name \rangle\}$

default: not used

Sets the *start value* and *options* for the *"first level"* continuing the numbering of the environment in which the $series=\{\langle series\ name\rangle\}$ key was executed. If passed *without value* this will only set *start value* continue the numbering from the last environment in which $series=\{\langle series\ name\rangle\}$ or $resume=\{\langle series\ name\rangle\}$ is not present and if the save-ans key is active it will continue the numbering from the last environment in which it was executed. The *start value* can be overwritten using the start key.

resume*

(value forbidden)

default: not used

Sets the *start value* and *options* for the *"first level"* continuing the numbering of the environment in which the $series=\{\langle series\ name \rangle\}$ or $resume=\{\langle series\ name \rangle\}$ keys are NOT present, if the save-ans key is active it will continue the numbering from the last environment in which it was executed. The *start value* can be overwritten using the start key.

© For security reasons the series key will never save in $\{\langle series\ name \rangle\}$ the keys series, resume, resume*, save-ans, save-key and start. When using the key resume= $\{\langle series\ name \rangle\}$ it will have hierarchy in the $\langle keys \rangle$ that are saved in $\{\langle series\ name \rangle\}$, in order to establish the value of a $\langle key \rangle$ already saved in $\{\langle series\ name \rangle\}$ it must be placed to the "right" of resume= $\{\langle series\ name \rangle\}$, the same thing happens with the resume* key, the exception is the save-ans key

that must be placed on the "left" if you want to start the numbering with its value. The resume key passed "without value" must be exactly "without value", i.e. resume= cannot be used and if executed before resume* it will affect the

4.5 Keys for multicols

```
columns = \{\langle integer \rangle\}
```

default: 1

Set the *number of columns* to be used by the multicols environment within the environment. The value must be a positive integer less than or equal to 10.

```
columns-sep = \{\langle rigid \ length \rangle\}
```

default: by level

Set the *space between* columns used by the multicols environment within the environment. Internally sets the value of \columnsep, by default its value is equal to the sum of the values set in the keys labelwidth and labelsep of the current level.

of The \footnote $\{\langle text \rangle\}$ command in the nested levels of multicols will not work as expected, prefer the use of $\five \five \fiv$ ment or via the after key.

4.6 Keys for minipage

```
mini-env = \{\langle rigid\ length\rangle\}
```

default: not used

Sets the width of the minipage environment on the "right side". This value added to the value set by the mini-sep key to determines the width of the minipage environment on the "left side", taking \linewidth as the maximum reference value.

```
mini-sep = \{\langle rigid \ length \rangle\}
```

default: 0.3333em

Sets the space between the minipage environment on the "left side" and the minipage environment on the "right side". This separation is applied together with \hfill.

4.6.1 The command \miniright

\miniright*

\miniright The \miniright command close the minipage environment on the "left side" and opens the minipage environment on the "right side" by starting it with the \centering command. It must be placed "after" the last \item of the current environment and "before" starting the material to be placed on the "right side". The starred version '*' inhibits the use of \centering command i.e. the usual LTFX justification is maintained in the minipage on the "right side".

of The \footnote $\{\langle text \rangle\}$ command in minipage environment will work as usual. If you prefer the footnotes to be numbered (not lowercase) and outside the environment, use $\lceil \text{footnotemark} \lceil \text{number} \rceil$ inside the environment and \footnotetext[$\langle number \rangle$] { $\langle text \rangle$ } outside the environment or via the after key.

4.6.2 The key miniright

In the horizontal list environments enumext* and keyans* it is not possible to use the \miniright command and the miniright key must be used instead.

```
miniright = \{ \langle code for drawing or tabular \rangle \}
```

default: not used

Set the code for the drawing or tabular to be placed in the minipage environment on the "right side" by starting it with \centering.

```
miniright^* = \{ \langle code \ for \ drawing \ or \ tabular \rangle \}
```

default: not used

Same as above, but without starting with \centering.

The storage system

The entire mechanism for "storing content" it is activated according to save-ans key on the "first level" of enumext or enumext* environments and it is ignored if they are established when they are nested inside each other. Only when this $\langle key \rangle$ is "active" the \anskey command and the environments keyans, keyans* and keyanspic are available.

```
\begin{enumext}[save-ans={\langle store\ name\rangle}]
                                                            \begin{enumext} [save-ans={ \langle store name \rangle}]
  \item Text
                                                               \item Text
    \begin{keyans}
                                                                 \begin{keyanspic}
    \end{keyans}
                                                                 \end{keyanspic}
\end{enumext}
                                                            \end{enumext}
```

5.1 Keys for storage

```
save-ans = \{ \langle store \ name \rangle \}
```

default: not set

Sets the name of the \(\sequence\) and \(\setaprop list\) in which the contents will be "stored" by \anskey in enumext and enumext* environments, \item* in keyans and keyans* environments and \anspic* in keyanspic environment. If the \(sequence \) or \(\sqrt{prop list} \) does not exist, it will be created globally and will not be overwritten if the key is used again.

```
wrap-ans = \{\langle code \{ \#1 \} \mid more \ code \rangle \}
```

default: \fbox{#1}

Wraps the *current argument* passed \anskey command to referenced by $\{\#1\}$. The $\{\langle code \rangle\}$ must be passed between braces and only affects the \(\current argument \) passed to \anskey and NOT the "stored

content" in the \(\store\) name\(\) set by save-ans key. If this key is passed using the \setenumext command it is necessary to use double '{##1}'.

 $wrap-opt = \{\langle code \{ \#1 \} \ more \ code \rangle \}$

Wraps the *optional argument* passed to the \item* and \anspic* commands referenced by {#1} in the keyans, keyans* and keyanspic environments. The $\{\langle code \rangle\}$ must be passed between braces and only affects the current *(optional argument)* and NOT the "stored content" in *(store name)* set by save-ans key. If this key is passed using the \setenumext command, it is necessary to use the double '{##1}'.

 $save-sep = \{ \langle text \ symbol \rangle \}$

Sets the *text symbol* that will separate the current $\langle label \rangle$ defined by the label key from the $\langle optional \rangle$ argument) (if present), when storing them in the (store name) defined by the save-ans key for the \item* command in the keyans and keyans* environment and for the \anspic command in the keyanspic environment. The $\{\langle text\ symbol \rangle\}$ must always be passed between braces, whitespace ' \sqcup ' is preserved within the braces and only affects the "stored content" and not what is displayed when using the show-ans or show-pos keys.

 $mark-ans = \{\langle symbol \rangle\}$

default: \textasteriskcentered

Sets the symbol to be displayed in the left margin of the "stored content" in \(\store\) name\(\) set by save-ans key when using show-ans key.

 $mark-pos = \{ \langle left \mid right \rangle \}$

Sets the aligned of the symbol defined by mark-ans key. The "symbol" is aligned in a box with the same dimensions of the label box defined by labelwidth key on the current level and separated by the value of the labelsep key.

5.2 Keys for internal label and ref

 $save-ref = \{ \langle true \mid false \rangle \}$

default: false

Activates the internal "label and ref" mechanism for referencing "stored content" in (store name) set by save-ans key. To reference the location of the "stored content" within the environment you must use $\mathsf{ref}\{\langle \mathit{store name} : \mathit{position} \rangle\}$, where $\langle \mathit{position} \rangle$ corresponds to the position occupied by the "stored content" in the *store name* returned by the show-pos key. For example \ref{test:4} will return 3. (b) which corresponds to the location of the "stored content" at position 4 within the environment in which the key save-ans=test was set.

 $mark-ref = \{\langle symbol \rangle\}$

default: \textasteriskcentered

Sets the *symbol* that will be displayed by the \printkeyans command only if the hyperref package is detected and the save-ref key are active. This "symbol" is used as a "link" between the environment in which the save-ans key was used and the place where the command is executed.

Keys for debugging and checking 5.3

 $\mathsf{show-ans} = \{ \langle \mathit{true} \mid \mathit{false} \rangle \}$

default: false

Displays the *current* \(\langle argument \rangle \text{ passed to \anskey in enumext environment, the current \(\langle label \rangle \text{ for } \) \item* in keyans environment and the current $\langle label \rangle$ for \anspic* in keyanspic environment at the place where it is executed. If the optional argument is present in \item* or \anspic* it will be shown in square brackets.

 $show-pos = \{\langle true \mid false \rangle\}$

Displays the position occupied by the "stored content" by \anskey in enumext environment, \item* in keyans environment and \anspic* in keyanspic environment in \(\store name\) set by save-ans key. This position is used by the \getkeyans command and by the \ref command if the save-ref key is active.

check-ans = $\{\langle true \mid false \rangle\}$

Enables the checking answer mechanism. This key works under the logic that each question will contain "only one answer", it is intended to be used in conjunction with no-store key.

no-store

default: not used

This is a meta-key that does not receive an argument. This key is used in conjunction with check-ans and is designed to be used with nested levels of enumext in which the \anskey command will not be used.

The command \anskey

 $\anskey \anskey{\langle content \rangle}$

The \anskey command takes a mandatory argument and is triggered by save-ans key. The "content" are "stored" in \(\store\) name\(\store\) set by save-ans key. The command does "not support" verbatim content and must NOT be nested. By design it is assumed that each \item or \item* will have a "single" occurrence of the command unless a nested level is opened or the no-store key is used. If save-ref key are active and the hyperref[7] package is detected, \hyperlink and \hypertarget will be used, otherwise the usual "label and ref" system provided by LTEX will be used.

Example

- ★ 1. Text containing our instructions or questions.
 - first answer
 - 2. Text containing our instructions or questions.
 - (a) Question.
 - second answer

```
3. Text containing our instructions or questions.
```

- third answer
- 4. Text containing our instructions or questions.
- fourth answer

```
\begin{enumext}[save-ans=test,show-ans=true]
  \item* Text containing our instructions or questions. \anskey\{\langle first \ answer \rangle\}
  \item Text containing our instructions or questions.
    \begin{enumext}
      \item Question.\anskey{\langle second answer\}}
    \end{enumext}
  \item Text containing our instructions or questions. \ankey{\langle third\ answer \rangle}
  \item Text containing our instructions or questions. \angle answer \
\end{enumext}
```

The environment keyans

```
\lceil \langle key = val \rangle \rceil  \item \item \[\langle custom \rangle \] \\item \[\langle content \rangle \] \\\ \end{keyans}
```

The keyans is an "enumerated list" environment designed for "multiple choice" questions activated by the save-ans key. This environment can NOT be nested and must always be at the "first level" of the enumext environment, the commands \item and \item [$\langle custom \rangle$] work in the usual.

```
\begin{enumext}[save-ans=test]
   \item \(\(\)item \(\)content\\)
      \begin{keyans} [\langle key = val \rangle]
          \item \(\(\)item \(\)content\\)
          \item [\langle custom \rangle] \langle item content \rangle
          \item* ⟨item content⟩
          \forall item^* [\langle content \rangle] \langle item content \rangle
      \end{keyans}
\end{enumext}
```

The \(\lambda keys\rangle\) set in the optional argument of the environment are the same (almost) as those of the enumext environment and have higher precedence than those set by $setenumext[\langle keyans \rangle] \{\langle key = val \rangle\}$. If the optional argument is not passed or the $\langle keys \rangle$ are not set by \setenumext, the default values will be the same as the second level of the enumext environment with the difference in the $\langle label \rangle$ which will be set to label=(\Alph^*).

5.5.1 The \item* in keyans

```
\item* \item*
```

The \item* and \item* [$\langle content \rangle$] command store the current $\langle label \rangle$ set by label key next to the $\langle content \rangle$ tent) (if it is present) in \(\store name \rangle \) set by save-ans key in the "first level" of the enumext environment.

The starred version '*' cannot be separated by spaces 'u' from the command, i.e. \item* and the optional argument does "not support" verbatim content. By design it is assumed that the starred version '*' will only appear "once" within the environment.

🂣 The behavior of \item* in keyans environment is NOT the same as in the enumext environment.

Example

```
\begin{enumext}[save-ans=test,columns=2,show-ans=true]
  \item Text containing a question.
   \begin{keyans}[nosep]
      \item Choice
      \item* Correct choice
      \item Choice
      \item Choice
    \end{keyans}
  \item Text containing a question and image.
    \begin{keyans}[nosep,mini-env={0.4\linewidth}]
      \item Choice
      \item Choice
      \item Choice
      \item Choice
      \times [(note)] Correct choice
      \miniright
```

```
\includegraphics[scale=0.25]{example-image-a}
      Some text
    \end{keyans}
\end{enumext}
```

- 1. Text containing a question.
 - (A) Choice
- * (B) Correct choice
 - (C) Choice
 - (D) Choice

- 2. Text containing a question and image.
 - (A) Choice
 - (B) Choice
 - (C) Choice
 - (D) Choice
- * (E) [note] Correct choice



Some text

The environment keyanspic

keyanspic \begin{keyanspic}[$\langle number\ above,\ number\ below$]\anspic{ $\langle drawing \rangle$ }\anspic*[$\langle content \rangle$]{ $\langle drawing \rangle$ }

The keyanspic is a "fake enumerated list" environment that which uses the \anspic command instead of \item. It is activated by the save-ans key and has the same settings as the keyans environment. It is intended for placing "drawings" or "tabular" with an in-line or above and below layout. A representation of the output can be seen in the figure 6.

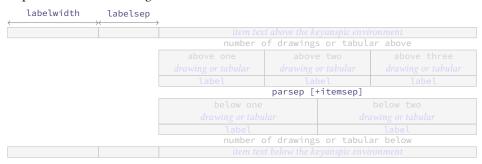


Figure 6: Representation of the keyanspic environment with optional argument [3,2] in enumext.

The optional argument determines the number drawings or tabular "above" and "below" within the environment. The vertical separation between "above" and "below" is controlled by the values set by parsep and itemsep keys passed to keyans environment. If the optional argument or the second part of it is omitted the drawings or tabular will be put on a single line.

5.6.1 The command \anspic

```
\anspic \anspic{\langle drawing \ or \ tabular \rangle}
                 \arrowvert anspic*[\langle content \rangle] \{\langle drawing \ or \ tabular \rangle\}
```

The \anspic command take three arguments, the *starred version* '*' store the current $\langle label \rangle$ next to the *(content)* (if it is present) in *(store name)* set by save-ans key.

The starred version '*' cannot be separated by spaces '_' from the command, i.e. \anspic* and the optional argument does "not support" verbatim content. By design it is assumed that the starred version '*' will only appear "once" within the environment.

Example

```
\begin{enumext}[save-ans=test,show-ans,nosep]
  \item Question with images.
   \begin{keyanspic}[3,2]
     \anspic{\includegraphics[scale=0.15]{example-image-a}}
     \anspic{\includegraphics[scale=0.15]{example-image-b}}
     \anspic{\includegraphics[scale=0.15]{example-image-a}}
     \anspic{\includegraphics[scale=0.15]{example-image-a}}
     \anspic*[note]{\includegraphics[scale=0.15]{example-image-a}}
    \end{keyanspic}
\end{enumext}
```

1. Question with images.











Printing stored content

The command \getkeyans

```
\getkeyans \getkeyans{\langle store name: position\rangle}
```

The command \getkeyans prints the "only stored content" in \(store name\)\) defined by save-ans key in the *(position)* returned by the show-pos key.

The "content" can only be accessed "after" it is stored, if the \(store name \) does not exist the command will return an error. The form taken by the argument *(store name : position)* is the same as that used to generate the internal "label and ref" system when save-ref key are active, so to refer to a stored "content". For example \getkeyans{test:4} will return the "stored content" at position 4 of the environment in which the key save-ans=test was set.

5.7.2 The command \printkeyans

```
\printkeyans \printkeyans [\langle keys \rangle] \{\langle store name \rangle\}
```

The command \printkeyans prints "all stored content" in {\store name\} defined by save-ans key. The "content" can only be accessed "after" it is stored, if \(\store name \) does not exist the command will return an error.

Internally it places the "stored content" inside the enumext environment with default values for label key are the same as those of the enumext environment along with the keys: nosep, first=\small, font=\small for all levels, except for the first one that adds the columns=2 key.

The optional argument allows to handle the \(\lambda \text{keys} \) "on the first level" of the enumext environment encapsulated by the command. If need to pass options for nested levels use $\setenumext[\langle print, level \rangle] \{\langle store, level \rangle\}$ $name \rangle \}.$

Example

```
\begin{enumext}[save-ans=sample,columns=2,show-pos=true,nosep,save-ref=true]
   \item Factor 3x+3y+3z. \anskey5(x+y+z)
   \item True False
     \begin{enumext}[nosep]
       \item \LaTeX2e\ is cool? \anskey{Very True!}
     \end{enumext}
   \item Related to Linux
     \begin{enumext}[nosep]
       \item You use linux? \anskey{Yes}
       \item Rate the following package and class
         \begin{enumext} [nosep]
           \item \texttt{xsim} \anskey{very good}
           \item \texttt{exsheets} \anskey{obsolete}
         \end{enumext}
     \end{enumext}
 \end{enumext}
 The answer to \ref{sample:4} is \getkeyans{sample:4} and the answers to
 all the worksheets are as follows:
 \printkeyans{sample}
1. Factor 3x + 3y + 3z.
                                                  [3] Yes
[1] | 3(x+y+z)
                                                 (b) Rate the following package and class
                                                         xsim
2. True False
                                                      [4] very good
  (a) LATEX2e is cool?
                                                         exsheets
   [2] Very True!
                                                      [5] obsolete
3. Related to Linux
  (a) You use linux?
```

The answer to 3.(b).i is very good and the answers to all the worksheets are as follows:

```
1. 3(x+y+z)
```

2. (a) Very True! 3. (a) Yes very good (b) i. ii. obsolete

Full examples

Here I will leave as an example some adaptations questions taken from TeX-SX. The examples are attached to this documentation and can be extracted from your PDF viewer or from the command line by running:

\$ pdfdetach -saveall enumext.pdf

and then you can use the excellent arara1 tool to compile them.

Example 1

Adapted from the response given by Enrico Gregorio in Squares for answer choice options and perfect alignment to mathematical answers 🖹.

1. La velocità di $1,00 \times 10^2$ m/s espressa in km/h è: 3. La velocità di $1,00 \times 10^2$ m/s espressa in km/h è:

A	36 km/h.	A	36 km/h.
В	360 km/h.	В	360 km/h.
С	27,8 km/h.	С	27,8 km/h.
D	$3,60 imes 10^8$ km/h.	D	$3,60 \times 10^8 \mathrm{km/h}$

2. In fisica nucleare si usa l'angstrom (simbolo: 1 Å = 4. In fisica nucleare si usa l'angstrom (simbolo: 1 Å = 4). 1×10^{-10} m) e il fermi o femtometro (1 fm = 1×10^{-10} m) e il fermi o femtometro (1 fm = 1×10^{-15} m). Qual è la relazione tra queste due unità di misura?

unità di misura?

A
$$1 \text{ Å} = 1 \times 10^5 \text{ fm}.$$
B $1 \text{ Å} = 1 \times 10^{-5} \text{ fm}.$
C $1 \text{ Å} = 1 \times 10^{-15} \text{ fm}.$

 1×10^{-15} m). Qual è la relazione tra queste due

4. A

$$\begin{array}{|c|c|c|c|c|} \hline \textbf{B} & 1 \, \text{Å} = 1 \times 10^{-5} \, \text{fm}. \\ \hline \textbf{C} & 1 \, \text{Å} = 1 \times 10^{-15} \, \text{fm}. \\ \hline \textbf{D} & 1 \, \text{Å} = 1 \times 10^{3} \, \text{fm}. \\ \hline \end{array}$$

A $1 \text{ Å} = 1 \times 10^5 \text{ fm}.$

D
$$1 \text{ Å} = 1 \times 10^3 \text{ fm}.$$

Example 2

1. B

Adapted from the response given by Florent Rougon in Multiple choice questions with proposed answers in random order — addition of automatic correction (cross mark) $\stackrel{\triangle}{=}$.

3. B

1. La velocità di $1{,}00 \times 10^2$ m/s espressa in km/h è:

2. A

```
A 36 km/h.
✓ B 360 km/h.
  C 27,8 km/h.
  D 3,60 \times 10^8 \, \text{km/h}.
```

2. In fisica nucleare si usa l'angstrom (simbolo: $1 \text{ Å} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) 1×10^{-15} m). Qual è la relazione tra queste due unità di misura?

```
\sqrt{A} 1 Å = 1 \times 10^5 \text{ fm}.
   B 1 \text{ Å} = 1 \times 10^{-5} \text{ fm}.
   C 1 \text{ Å} = 1 \times 10^{-15} \text{ fm}
   D 1 \text{ Å} = 1 \times 10^3 \text{ fm}.
```

3. La velocità di $1{,}00 \times 10^2$ m/s espressa in km/h è:

```
A 36 km/h.
✓ B 360 km/h.
  C 27.8 \,\text{km/h}.
  D 3,60 \times 10^8 \,\text{km/h}.
```

4. In fisica nucleare si usa l'angstrom (simbolo: $1 \text{ Å} = 1 \times 10^{-10} \text{ m}$) e il fermi o femtometro (1 fm = 1×10^{-15} m). Qual è la relazione tra queste due unità di misura?

```
\sqrt{A} 1 Å = 1 \times 10^5 \text{ fm}.
    B 1 \text{ Å} = 1 \times 10^{-5} \text{ fm}.
    C 1 \text{ Å} = 1 \times 10^{-15} \text{ fm}
    D 1 \text{ Å} = 1 \times 10^3 \text{ fm}.
1. B
```

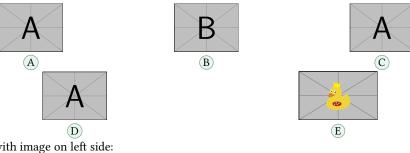
2. A 3. B

¹The cool T_EX automation tool: https://www.ctan.org/pkg/arara

Example 3

- A "simple multiple choice" test 🖹.
- 1. First type of questions
 - (A) value
 - (B) correct
 - (C) value
 - (D) value
- 2. Second type of questions
 - $2\alpha + 2\delta = 90^{\circ}$
 - II. $\alpha = \delta$
 - III. $\angle EDF = 45^{\circ}$
 - (A) I only
 - (B) II only
 - © I and II only
- 3. Third type of questions
 - (1) $2\alpha + 2\delta = 90^{\circ}$
 - (2) $\angle EDF = 45^{\circ}$
 - (A) value
 - (B) value
 - (C) value
- 4. Question with image and label below:

- (D) I and III only
- (E) I, II, and III
- (D) value
- (E) value



- 5. Question with image on left side:
 - (A) value
 - (B) value
 - (C) value
 - (D) correct
 - **E** value
- Test keys
- 1. B, x = 5
- 2. D
- 3. C, some note

- * 4. E, A duck
- * 5. D, other note

Example 4

A "simple worksheet" using ducks :) 🖹.



Factor $x^2 - 2x + 1$



Factor 3x + 3y + 3z

The following questions need to be cuaqtified:)



- True False
 - (a) $\alpha > \delta$
 - (b) LaTeX2e is cool?



Related to Linux

- (a) You use linux?
- (b) Usually uses the package manager?
- (c) Rate the following package and class
 - i. xsim-exam
 - ii. xsim
 - iii. exsheets

The answer to 1 is $(x-1)^2$ and the answer to 3.(a) is False.

- 1. $(x-1)^2$
- 2. 3(x+y+z)
- 3. (a) False
- (b) Very True!
- 4. (a) Yes

- (b) Yes, dnf
- (c) i. doesn't exist for now :(
- ii. very good
- iii. obsolete

Example 5

Adapted from the response given by Stephen in SAT like question format 🖹.

1

Which choice best describes what happens in the passage?

- A) One character argues with another character who intrudes on her home.
- B) One character receives a surprising request from another character.
- C) One character reminisces about choices she has made over the years.
- D) One character criticizes another character for pursuing an unexpected course of action.

2

Which choice best describes what happens in the passage?

- A) One character argues with another character who intrudes on her home.
- B) One character receives a surprising request from another character.
- C) One character reminisces about choices she has made over the years.
- D) One character criticizes another character for pursuing an unexpected course of action

3

Which choice best describes what happens in the passage?

- A) One character argues with another character who intrudes on her home.
- B) One character receives a surprising request from another character.
- C) One character reminisces about choices she has made over the years.
- One character criticizes another character for pursuing an unexpected course of action

4

Which choice best describes what happens in the passage?

- A) One character argues with another character who intrudes on her home.
- B) One character receives a surprising request from another character.
- C) One character reminisces about choices she has made over the years.
- D) One character criticizes another character for pursuing an unexpected course of action.

1. A)

2. C)

3. B)

4. D)

7 The way of non-enumerated lists

It is possible to use (or abuse) the enumext environment to mimic *non-enumerated* list environments such as itemize and description, clearly the $\langle keys \rangle$ to "store answers", the keyans and keyanspic environments lose their sense and it is not the focus of the main of this package, but, why not to do it?.

Here I leave as an example other uses of the enumext environment that can be helpful for specific purposes. The "trick" to generate these fake environments is set label= $\{\$ or label= $\{\$ and play with the list-indent, list-offset, font and wrap-label keys.

Fake itemize environment

Here we set the label key using the default settings in LTEX for the four levels \textbullet, \textendash, \textasteriskcentered and \textperiodcentered together with the nosep key to reduce the vertical spaces in the left side example and set the label key in mathematical mode for the right side as \ast, \diamond, \circ and \star for the four levels together with the nosep key

- First level item
 - Second level item
 - * Third level item
 - · Fourth level item
- · First level item

- * First level item
 - ⋄ Second level item
 - Third level item
 - * Fourth level item
- * First level item

Fake description environment

Here we set label={} and list-indent=2.5em, font=\bfseries.

SomeThing A short one-line description.

This is an entry without a label.

Something A short *one-line* description text.

Something long A much *longer* description text may take more than one line or more than one paragraph. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

If we add list-indent=Opt you get widest style:

SomeThing A short one-line description.

This is an entry without a label.

Something A short *one-line* description text.

Something long A much *longer* description text may take more than one line or more than one paragraph. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

The small space at the beginning of the "unlabeled entry" corresponds to \labelsep and can be removed using \hspace{-\labelsep} at the beginning of the line.

Description indented by label

Here we set label={} and we will give a convenient value to labelsep and labelwidth, for example we can take as reference our *longest label* and pass it as value using:

```
\newlength{\descitemwd}
\settowidth{\descitemwd}{\textbf{Something long}}
```

and then use labelsep=4pt, labelwidth=\descitemwd, font=\bfseries.

SomeThing A short one-line description.

This is an entry without a label.

Something A short one-line description.

Something long A much longer description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut

purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida

mauris.

The environment can be translated so that the $\langle labels \rangle$ are on the left margin calculating the value passed to the list-offset key, in this case it will be equal to the sum of the values set by the labelwidth and labelsep keys finally resulting as list-offset={-\descitemwd - 4pt}.

SomeThing A short one-line description.

This is an entry without a label.

Something A short one-line description.

Something long A much longer description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

If we add align=right it will look like this:

SomeThing A short one-line description.

This is an entry without a label.

Something A short one-line description.

Something long A much longer description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

At this point we have used list-offset={-\descitemwd - 4pt} instead of list-offset={-\labelwidth - \labelsep}, this is because the parameters \labelwidth and \labelsep take the default values, as if we had not set label.

Description with multi-line labels

The label key does not accept *multiline material*, this is where the wrap-label* key comes into play. Unlike the enumitem package, the align key only supports three options, so what we will do is create a command in the style \parleft of enumitem that allows us to place *multiline labels* using \parbox.

```
\NewDocumentCommand \itembx { s +m }
    {%
     \IfBooleanTF{#1}
        {\strut\smash{\parbox[t]{\labelwidth}{\raggedright{#2}}}}%
        {\strut\smash{\parbox[t]{\labelwidth}{\raggedleft{#2}}}}%
}
```

Now we just need to set $wrap-label*={\langle itembx\{#1\} \rangle}.$

SomeThing A short one-line description.

This is an entry without a label.

Something A short one-line description.

Something A much longer description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, **long** vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

SoMeThInG A much longer description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, **LoNg** vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

Final notes

The original implementation (if you can call it that) of the ideas that led to the creation of enumext were some macros using the enumerate[4] package for personal use created in early 2003, the code was quite questionable, but functional for these simple requirements.

With the great answers given by Christian Hupfer in Create a fake label ref using list and the answer given by David Carlisle in Change the use of label ref by data save in an array (list) I managed to create a more solid code than the original version, now using the <code>l3prop[10]</code> and <code>l3seq[10]</code> modules together with the <code>hyperref[7]</code> and <code>enumitem[5]</code> packages, which did the job, but with some limitations.

As time went by I took these limitations as a personal challenge which I called "reinventing the wheel", since there were packages and classes that did more or less what I was looking for, but did not fit my simple requirements. This "reinventing the wheel" finally ended up becoming enumext.

Why list environments?

The answer is simple, first I love the beauty of its syntax and many of what I had already written used the enumerate environment or lists created using the enumitem package. In my mind I thought: how complicated could it be to write a package that looked like enumitem? It seemed simple enough, of course I didn't have in mind the mess I was getting into working with list environments, minipage and adding support for the multicol and hyperref packages.

Of course, seeing the final result of the experiment "reinventing the wheel" I am quite satisfied.

Why not random questions and other utilities

The "random" type questions I love and hate them at the same time, although they simplify a lot the work when creating a multiple choice test, but you lose the beauty of typessetting a document with ETeX, that is to say the output does not always look as nice as it should, even if they are only alternatives these must follow a certain order when presented either numerical or presentation, that said handling that using nested lists is quite complicated so I do not classify to be implemented.

8 References

- [1] HIRSCHHORN, PHILIP. "Using the exam document class". Available from CTAN, https://www.ctan.org/pkg/exam, 2023.
- [2] NIEDERBERGER, CLEMENS. "xsim eXercise Sheets IMproved". Available from CTAN, https://www.ctan.org/pkg/xsim, 2023.
- [3] MITTELBACH, FRANK. "An environment for multicolumn output". Available from CTAN, https://www.ctan.org/pkg/multicol, 2024.
- [4] The LaTeX Project. "enumerate Enumerate with redefinable labels". Available from CTAN, https://www.ctan.org/pkg/enumerate, 2024.
- [5] Bezos, Javier. "Customizing lists with the enumitem package". Available from CTAN, https://www.ctan.org/pkg/enumitem, 2019
- [6] Berry, Karl. "ΜΈχ 2_ε: An Unofficial Reference Manual". Available from ctan, https://ctan.org/pkg/latex2e-help-texinfo, 2024.
- [7] The LTEX Project. "Extensive support for hypertext in LTEX". Available from CTAN, https://www.ctan.org/pkg/hyperref, 2024.
- [8] Burnol, Jean-François. "The footnotehyper package". Available from CTAN, https://www.ctan.org/pkg/footnotehyper, 2021.
- [9] The LATEX Project. "The expl3 package". Available from CTAN, https://www.ctan.org/pkg/l3kernel, 2024.
- [10] The LTeX Project. "The LTeX3 Interfaces". Available from CTAN, https://www.ctan.org/pkg/l3kernel, 2024.
- [11] The LaTeX Project. "The xparse package". Available from CTAN, https://www.ctan.org/pkg/xparse, 2024.
- [12] GUNDLACH, PATRICK. "The lua-visual-debug package". Available from CTAN, https://www.ctan.org/pkg/lua-visual-debug, 2023.
- [13] Lemvig, Mogens. "The shortlst package". Available from ctan, https://www.ctan.org/pkg/shortlst, 1998.
- [14] NIEDERBERGER, CLEMENS. "tasks Horizontally columned lists". Available from CTAN, https://www.ctan.org/pkg/tasks, 2022.

9 Change history

v1.0 2024-05-26 - First public release.

Index of Documentation 10

The italic numbers denote the pages where the corresponding entry is described.

C	below 8
Document class:	check-ans
article 2	columns-sep 4, 10
book	columns 4, 8, 10
exam 2	first 9
letter 2	font 7
report 2	item-pos* 5, 6
\columnbreak 4	item-sym* 5, 6
\columnsep 10	itemindent 8
Commands provide by enumext:	itemsep 8, 13
\anskey 4, 10-12	labelsep 3, 5-11, 18
\anspic* 4, 10, 11, 13	labelwidth 3, 6, 7, 9-11, 18
\anspic 11, 13	labelwith 5
\getkeyans	label 7, 9, 11, 12, 14, 17, 18
\item* 4-7, 10-12	list-indent
\itemwidth 5 \item 5-7, 9-12	list-offset
\miniright	listparindent9 mark-ans11
\printkeyans 4, 6, 11, 14	mark-pos
\setenumext	mark-ref
Counters defined by enumext:	mini- right* 6
enumXiii4	mini-env
enumXii4	mini-right
enumXiv 4	mini-sep 4, 10
enumXi4	miniright* 10
enumXviii 4	miniright
enumXvii4	no-store
enumXvi4	noitemsep 8
enumXv 4	nosep 8, 17
E	parsep 8, 13
Environments provide by enumext:	partopsep 8
enumext* 4-10	ref 4,7
enumext	resume*9
keyans*	resume*
keyanspic 4, 6, 8, 10, 11, 13, 17	resume
keyans 4-13, 17	save-ans
Environments:	save-key9
enumerate 1, 3, 5, 19	save-ref
figure5	save-sep
list 3, 9, 19	series 6, 9
minipage 3–5, 10, 19	show-ans11
multicols 3, 4, 10	show-length 7
table 5	show-pos
task 5	start9
F	topsep 8
\footnote 5	widest7
	wrap-ans 10
I	wrap-label* 7, 18
\item 3, 5	wrap-label 7
\itemsep 8	wrap-opt11
K	L
Keys for environments provide by enumext:	\label 4
above* 8	Labels provide by enumext:
above 8	\Alph* 7, 12
after 9, 10	\Roman* 7
align	\alph* 7
before* 9	\arabic* 7
before 9 below* 8	\roman*
	(tabetsep 3, 7
©2024 by Pablo González L	

\labelwidth	multicol 1, 2, 4, 19 task 5, 6 xsim 2
P Packages:	\parsep
enumerate	R \raggedcolumns
l3prop	T \topsep

11 Implementation

The most recent publicly released version of enumext is available at CTAN: https://www.ctan.org/pkg/enumext. While general feedback via email is welcomed, specific bugs or feature requests should be reported through the issue tracker: nhttps://github.com/pablgonz/enumext/issues.

The documentation presented here is far from professional, it contains a lot of obvious information that to the eye of a TeXpert are superfluous, but, after so many years developing this project is the only way to remember what does what

11.1 General conventions

Variables containing i, ii, iii and iv are associated by level with the enumext environment, variables containing v are associated with the keyans environment, variables containing vi are associated with the keyanspic environment, variables containing vii are associated with the enumext* environment and variables containing viii are associated with the keyans* environment.

To simplify writing and documentation some variables and functions that are common to the different levels of the environments are described using a capital "X".

The temporary function __enumext_tmp:n is used in different parts of the package code for variable creation or execution of other functions that are grouped into this one.

All variables and functions defined in this package are private and are NOT intended to work or be used by another package or module.

11.2 Initial set up

Start the DocStrip guards.

```
*package
```

Identify the internal prefix (LTFX3 DocStrip convention) for l3doc class.

```
2 (@@=enumext)
```

11.3 Declaration of the package

First we will make sure we have a minimum (super updated) version of ETFX to work correctly.

```
3 \NeedsTeXFormat{LaTeX2e} [2023-11-01]
```

Now declare the enumext package.

```
4 \ProvidesExplPackage
5 {enumext}
6 {2024-05-26}
7 {1.0}
8 {Enumerate exercise sheets}
```

Finally check if the multicol package is loaded, if not we load it.

11.4 Definition of variables

Variables that do not appear in this section are created by means of \keys_define:nn or some function described below.

\l__enumext_level_int
\l__enumext_keyans_level_int
\l__enumext_keyans_level_hint
\l__enumext_keyans_level_hint
\l__enumext_keyans_pic_level_int
\l__enumext_starred_bool
\l__enumext_starred_bool
\l__enumext_starred_first_bool
\l__enumext_standar_bool
\l__enumext_standar_bool
\l__enumext_standar_first_bool
\l__enumext_keyans_env_bool

\l__enumext_keyans_env_bool

\l__enumext_level_int
\l__enumext_starred_hint
\l__enumext_starred_bint
\l__enumext_numext_starred_bint
\l__enumext_numext_starred_bint
\l__enumext_num

Integer variables will control the nesting levels of the environments and boolean variables will be used to determine if they are present (nested) in each other. The boolean variables $\g_{enumext_starred_bool}$ and $\g_{enumext_standar_bool}$ will be set to "true" when the enumext and enumext* environments are not nested with each other.

```
20 \int_new:N \l__enumext_level_int
21 \int_new:N \l__enumext_level_h_int
22 \int_new:N \l__enumext_keyans_level_int
23 \int_new:N \l__enumext_keyans_level_h_int
24 \int_new:N \l__enumext_keyans_pic_level_int
25 \bool_new:N \l__enumext_starred_bool
26 \bool_new:N \g_enumext_starred_bool
```

```
27 \bool_new:N \l__enumext_starred_first_bool
                                 _{28} \bool_new:N \l__enumext_standar_bool
                                 29 \bool_new:N \g__enumext_standar_bool
                                 30 \bool_new:N \l__enumext_standar_first_bool
                                 31 \bool_new:N \l__enumext_keyans_env_bool
                                (End of definition for \l_{-}enumext_level_int and others.)
                               Variables to store the "name of the counters" enumXi, enumXii, enumXiii and enumXiv for enumext
    \l enumext counter i tl
                                environment, enumXv for keyans environment and enumXvi for the keyanspic environment.
   \l__enumext_counter_ii_tl
                                The counters enumXviii and enumXviii are used by enumext* and keyans* environments.
  \l__enumext_counter_iii_tl
                                The initial values of these variables are set by the function \__enumext_define_counters: Nn (§11.8)
   \l__enumext_counter_iv_tl
                                and then modified by the function \__enumext_label_style: Nnn used by label key (§11.11).
    \l enumext counter v tl
   \l__enumext_counter_vi_tl
                                 32 \cs_set_protected:Npn \__enumext_tmp:n #1
  \l__enumext_counter_vii_tl
 \l__enumext_counter_viii_tl
                                       \tl_new:c { l__enumext_counter_#1_tl }
                                34
                                 35
                                 36 \clist_map_inline:nn { i, ii, iii, iv, v, vi, viii } { \__enumext_tmp:n {#1} }
                                (End of definition for \l_enumert_counter_i_tl and others.)
\c_enumext_counter_style_tl Internal variables used by ref key (§11.11).
 \l__enumext_ref_key_arg_tl
                                 37 \tl_const:Nn \c__enumext_counter_style_tl
\l__enumext_ref_the_count_tl
                                38 { { arabic } { roman } { Roman } { alph } { Alph } }
                                 39 \tl_new:N \l__enumext_ref_key_arg_tl
\l__enumext_the_counter_X_tl
                                 40 \tl_new:N \l__enumext_ref_the_count_tl
     \l__enumext_renew_the_count_X_tl
                                 \cs_set_protected:Npn \__enumext_tmp:n #1
                                 42 {
                                       \tl_new:c { l__enumext_renew_the_count_#1_tl }
                                 43
                                       \tl_new:c { l__enumext_the_counter_#1_tl }
                                       \tl_set:ce { l__enumext_the_counter_#1_tl } { \exp_not:c { theenumX#1 } }
                                 47 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { \__enumext_tmp:n {#1} }
                                (End of definition for \c_-enumext_counter_style_tl and others.)
      \g__enumext_resume_int Internal variables used by resume, resume* and series keys. The global token list \g__enumext_-
  \g__enumext_resume_vii_int item_symbol_tlis used by item-sym* key (§11.26).
  \l__enumext_resume_name_tl
                                48 \int_new:N \g__enumext_resume_int
      \l__enumext_resume_active_bool
                                49 \int_new:N \g__enumext_resume_vii_int
  \g__enumext_item_symbol_tl
                                50 \tl_new:N
                                                \l__enumext_resume_name_tl
                                51 \bool_new:N \l__enumext_resume_active_bool
       \g__enumext_standar_series_tl
                                52 \tl_new:N \g__enumext_item_symbol_tl
       \g__enumext_starred_series_tl
                                               \g__enumext_standar_series_tl
                                 53 \tl_new:N
                                 54 \tl_new:N \g__enumext_starred_series_tl
                                (End of definition for \g_{\text{enumext\_resume\_int}} and others.)
                                The variable \l__enumext_current_widest_dim stores the current label width, the variable \g__-
       \l__enumext_current_widest_dim
                                enumext_counter_styles_tl stores the default \(\lambda label style\rangle\) and the variable \(\gramge_\)enumext_widest_-
       \g__enumext_counter_styles_tl
                                label_tl the label width. These variables are used by widest (§11.12) and label (§11.10) keys.
 \g__enumext_widest_label_tl
      \l__enumext_label_width_by_box
                                 55 \dim_new:N \l__enumext_current_widest_dim
                                 56 \tl_new:N \g__enumext_counter_styles_tl
                                 57 \tl_new:N \g__enumext_widest_label_tl
                                 58 \box_new:N \l__enumext_label_width_by_box
                                (End of definition for \l__enumext_current_widest_dim and others.)
                               The boolean variable \l__enumext_leftmargin_tmp_X_bool and the dimensional variable \l__-
    \l__enumext_leftmargin_tmp_X_bool
                                enumext_leftmargin_tmp_X_dim are used by the list-indent key (§11.14).
     \l__enumext_leftmargin_tmp_X_dim
\l__enumext_leftmargin_X_dim
                                The variables \l__enumext_leftmargin_X_dim and \l__enumext_itemindent_X_dim are used (and
\l__enumext_itemindent_X_dim
                                set) by the function \__enumext_calc_hspace: NNNNNNNNNNN (§11.30.1) which determines the internal
                                values for \leftmargin and \itemindent.
                                 59 \cs_set_protected:Npn \__enumext_tmp:n #1
                                        \bool_new:c { l__enumext_leftmargin_tmp_#1_bool }
                                       \dim_new:c { l__enumext_leftmargin_tmp_#1_dim }
                                       \dim_new:c { l__enumext_leftmargin_#1_dim
                                 63
                                       \dim_new:c { l__enumext_itemindent_#1_dim
                                 66 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { \__enumext_tmp:n {#1} }
```

©2024 by Pablo González L

(End of definition for $\l_enumext_leftmargin_tmp_X_bool$ and others.)

\l__enumext_multicols_above_X_skip
\l__enumext_multicols_below_X_skip

Internal variables used by columns key §11.18).

```
67 \cs_set_protected:Npn \__enumext_tmp:n #1
68 {
69    \skip_new:c { l__enumext_multicols_above_#1_skip }
70    \skip_new:c { l__enumext_multicols_below_#1_skip }
71  }
72 \clist_map_inline:nn { i, ii, iii, iv, v } { \__enumext_tmp:n {#1} }
```

 $(\textit{End of definition for } \verb|\|l_enumext_multicols_above_X_skip| and \verb|\|l_enumext_multicols_below_X_skip|)$

\g__enumext_minipage_stat_int
\l__enumext_minipage_left_skip
\l__enumext_minipage_right_skip
\l__enumext_minipage_after_skip
\g__enumext_minipage_right_skip
\g__enumext_minipage_after_skip
\l__enumext_minipage_left_X_dim
\l__enumext_minipage_active_X_bool

Internal variables used by \miniright command (§11.19.4) and the keys miniright, miniright*, minienv and mini-sep (§11.17, §11.19).

```
73 \int_new:N \g__enumext_minipage_stat_int
74 \skip_new:N \l__enumext_minipage_left_skip
75 \skip_new:N \l__enumext_minipage_right_skip
76 \skip_new:N \l__enumext_minipage_after_skip
77 \skip_new:N \g__enumext_minipage_right_skip
78 \skip_new:N \g__enumext_minipage_after_skip
79 \cs_set_protected:Npn \__enumext_tmp:n #1
80 {
81    \dim_new:c { l__enumext_minipage_left_#1_dim }
82    \bool_new:c { l__enumext_minipage_active_#1_bool }
83  }
84 \clist_map_inline:nn { i, ii, iii, iv, v, vii, viii } { \__enumext_tmp:n {#1} }
```

(End of definition for \g_{-} enumext_minipage_stat_int and others.)

\l_enumext_wrap_label_X_bool
\l_enumext_wrap_label_opt_X_bool
\l_enumext_start_X_int
\l_enumext_fake_item_indent_X_tl
\l_enumext_label_fill_left_X_tl
\l_enumext_label_fill_right_X_tl
\l_enumext_vspace_a_star_X_bool
\l_enumext_vspace_b_star_X_bool

The integer variable \l__enumext_start_X_int are used by the start key (§11.12), the token list \l__enumext_fake_item_indent_X_tl is used by itemindent key, the variables \l__enumext_label_fill_left_X_tl are used by the align key (§11.10). The boolean vars \l__enumext_vspace_a_star_X_bool, \l__enumext_vspace_b_star_X_bool are used by above, above*, below and below* keys

```
85 \cs_set_protected:Npn \__enumext_tmp:n #1
86
      \bool_new:c { l__enumext_wrap_label_#1_bool
87
      \bool_new:c { l__enumext_wrap_label_opt_#1_bool }
88
      \int_new:c { l__enumext_start_#1_int
                 { l__enumext_fake_item_indent_#1_tl }
      \tl_new:c
      \tl new:c
                  { l__enumext_label_fill_left_#1_tl
      \tl new:c
                  { l__enumext_label_fill_right_#1_tl }
      \bool_new:c { l__enumext_vspace_a_star_#1_bool
      \bool_new:c { l__enumext_vspace_b_star_#1_bool }
<code>% \clist_map_inline:nn { i, ii, iii, iv, v, vii, viii } { \__enumext_tmp:n {#1} }</code>
```

(End of definition for $\l_enumext_wrap_label_X_bool$ and others.)

\l_enumext_store_active_bool
\l_enumext_store_name_tl
\g_enumext_store_name_tl
\l_enumext_store_anskey_arg_tl
\l_enumext_store_columns_join_int
\l_enumext_store_keyans_label_tl
\l_enumext_store_keyans_item_opt_tl
\l_enumext_keyans_item_opt_tl
\l_enumext_keyans_tmpa_tl

The boolean variable \l_enumext_store_active_bool setting by save-ans key (§??) activates all the mechanism related to \anskey, keyans, keyans* and keyanspic.

The variable \l__enumext_store_name_tl sets the name for the storage in $\langle sequence \rangle$ and $\langle prop \ list \rangle$, the variable \g__enumext_store_name_tl is just a copy of the storage name used by the check-ans key (§??).

The variable \l__enumext_store_anskey_arg_tl stores the contents of \anskey ($\S11.24$) and the variable \l__enumext_store_keyans_label_tl stores the contents of \item* ($\S11.28.2$) for the keyans and keyans* environments and the contents of \anspic* ($\S11.33.1$) for the keyanspic environment.

The variable \l__enumext_keyans_tmpa_tl is a temporary variable used by keyans and keyanspic at various points.

```
97 \bool_new:N \l__enumext_store_active_bool
98 \tl_new:N \l__enumext_store_name_tl
99 \tl_new:N \g__enumext_store_name_tl
100 \tl_new:N \l__enumext_store_anskey_arg_tl
101 \int_new:N \l__enumext_store_columns_join_int
102 \tl_new:N \l__enumext_store_keyans_label_tl
103 \tl_new:N \l__enumext_store_keyans_item_opt_tl
104 \tl_new:N \l__enumext_keyans_item_opt_tl
105 \tl_new:N \l__enumext_keyans_tmpa_tl
```

©2024 by Pablo González L

24/125

```
(End of definition for \l_enumert_store_active_bool and others.)
                                 Internal variables used by the command \setenumext (§11.38).
  \l__enumext_setkey_tmpa_tl
  \l__enumext_setkey_tmpb_tl
                                  106 \tl_new:N \l__enumext_setkey_tmpa_tl
 \l__enumext_setkey_tmpa_int
                                  107 \tl_new:N \l__enumext_setkey_tmpb_tl
 \l__enumext_setkey_tmpa_seq
                                  108 \int_new:N \l__enumext_setkey_tmpa_int
                                  \seq_new:N \l__enumext_setkey_tmpa_seq
 \l__enumext_setkey_tmpb_seq
                                  \seq_new:N \l__enumext_setkey_tmpb_seq
                                 (End of definition for \l_enumext_setkey_tmpa_tl and others.)
                                 Internal variables used by [\langle key = val \rangle] in enumext and enumext* environment, the command
   \l__enumext_print_keyans_starred_tl
                                 \printkeyans (§11.37) and save-key key.
      \l__enumext_store_save_key_X_tl
       \l__enumext_print_keyans_X_tl
                                  \tl_new:N \l__enumext_print_keyans_starred_tl
  \l__enumext_store_upper_level_X_bool
                                  \cs_set_protected:Npn \__enumext_tmp:n #1
                                  113
                                         \tl_new:c { l__enumext_store_save_key_#1_tl
                                  114
                                         \bool_new:c { l__enumext_store_save_key_#1_bool }
                                         \tl_new:c { l__enumext_store_active_keys_#1_tl }
                                  116
                                         \tl_new:c
                                                      { l__enumext_print_keyans_#1_tl
                                         \bool_new:c { l__enumext_store_upper_level_#1_bool }
                                  118
                                  119
                                  120 \clist_map_inline:nn { i, ii, iii, iv, vii } { \__enumext_tmp:n {#1} }
                                 (End\ of\ definition\ for\ \l_enumext\_print\_keyans\_starred\_tl\ and\ others.)
                                 Internal variables for "storage system" mechanism used by \anskey (§11.24), keyans and keyanspic
\l__enumext_show_answer_bool
                                 environments. These variables are used by show-ans, show-pos, mark-ans, save-key and mark-ref
       \l__enumext_show_position_bool
 \l__enumext_mark_ref_sym_tl
                                 keys (§11.23).
       \l__enumext_mark_answer_sym_tl
                                  \text{\lool_new:N \l__enumext_show_answer_bool}
        \l__enumext_mark_position_str
                                  \bool_new:N \l__enumext_show_position_bool
                                  123 \tl_new:N \l__enumext_mark_ref_sym_tl
                                  124 \tl_new:N \l__enumext_mark_answer_sym_tl
                                  \str_new:N \l__enumext_mark_position_str
                                 (End\ of\ definition\ for\ \l_enumext\_show\_answer\_bool\ and\ others.)
                                 Internal variables used by keyanspic environment (§11.33.2).
      \l__enumext_keyans_pic_body_seq
     \l__enumext_keyans_pic_width_dim
                                  \seq_new:N \l__enumext_keyans_pic_body_seq
     \l enumext kevans pic above int
                                  127 \dim_new:N \l__enumext_keyans_pic_width_dim
     \l__enumext_keyans_pic_below_int
                                  128 \int_new:N \l__enumext_keyans_pic_above_int
                                  \int_new:N \l__enumext_keyans_pic_below_int
     \l__enumext_keyans_pic_above_skip
                                  \skip_new:N \l__enumext_keyans_pic_above_skip
                                 (End of definition for \l_enumext_keyans_pic_body_seq and others.)
                                 Internal variables used by "check answer" mechanism (§11.22.3) used by the check-ans and no-store
       \l__enumext_check_answers_bool
                                 keys and check for starred commands \item* in keyans and keyans* environments and \anspic* in
       \l__enumext_check_ans_key_bool
                                 keyanspic environment.
       \g__enumext_check_ans_key_bool
   \l enumext check start line env tl
                                  \text{\lool_new:N \l__enumext_check_answers_bool
   \g__enumext_start_line_tl
                                  132 \bool_new:N \l__enumext_check_ans_key_bool
    \g__enumext_check_starred_cmd_int
                                  \dool_new:N \g__enumext_check_ans_key_bool
                                  134 \tl_new:N \l__enumext_check_start_line_env_tl
 \g__enumext_item_anskey_int
                                  135 \tl_new:N
                                                 \g__enumext_start_line_tl
 \g__enumext_item_number_int
                                  136 \tl_new:N
                                                 \g__enumext_envir_name_tl
                                  137 \int_new:N \g__enumext_check_starred_cmd_int
                                  138 \int_new:N \g__enumext_item_anskey_int
                                  139 \int_new:N \g__enumext_item_number_int
                                  140 \int_new:N \g__enumext_item_answer_diff_int
                                 (End of definition for \l_enumext_check_answers_bool and others.)
   \l__enumext_hyperref_bool
                                 The boolean variable \l__enumext_hyperref_bool will determine if the hyperref package is present
                                 or load in memory (§11.7). The boolean variable \l__enumext_footnotes_key_bool determine if
       \l__enumext_footnotes_key_bool
                                  hyperref is load with key hyperfootnotes=true.
                                  \text{\text{bool_new:N \l__enumext_hyperref_bool}
                                  {}_{^{142}}\ \backslash bool\_new:N\ \backslash l\_\_enumext\_footnotes\_key\_bool
                                 (\textit{End of definition for} \ \backslash \ l\_\_enumext\_hyperref\_bool \ \ and \ \backslash \ l\_\_enumext\_footnotes\_key\_bool.)
```

©2024 by Pablo González L 25/125

```
\l__enumext_newlabel_arg_one_tl
      \l__enumext_newlabel_arg_two_tl
  \l__enumext_store_write_aux_file_tl
\l__enumext_label_copy_X_tl
```

Internal variables are used when executing the save-ref key. The variables $\lower label_$ copy_X_tl correspond to temporary copies of the labels defined by level on which operations will be performed.

be used to form the arguments passed to the function __enumext_newlabel:nn and the variable \l__enumext_store_write_aux_file_tl will be in charge of executing the writing code in the .aux file.

```
143 \tl_new:N \l__enumext_newlabel_arg_one_tl
\tl_new:N \l__enumext_newlabel_arg_two_tl
145 \tl_new:N \l__enumext_store_write_aux_file_tl
146 \cs_set_protected:Npn \__enumext_tmp:n #1
    \tl_new:c { l__enumext_label_copy_#1_tl }
149
```

($End\ of\ definition\ for\ \l_enumext_newlabel_arg_one_tl\ and\ others.$)

\g enumext footnote int \g__enumext_footnote_arg_seq \g__enumext_footnote_int_seq

Internal variables used for redefinition of \footnote.

```
_{151} \int_new:N \g__enumext_footnote_int
\seq_new:N \g__enumext_footnote_arg_seq
\seq_new:N \g__enumext_footnote_int_seq
```

\l__enumext_item_starred_X_bool l enumext item column pos X int \g__enumext_item_count_all_X_int \l__enumext_joined_item_X_int \l__enumext_joined_item_aux_X_int \l__enumext_tmpa_X_int \l__enumext_item_text_X_box \l__enumext_joined_width_X_dim \l__enumext_item_width_X_dim \g__enumext_item_symbol_aux_X_tl \l__enumext_align_label_X_str \g__enumext_minipage_active_X_bool \g__enumext_miniright_code_X_tl \g__enumext_minipage_center_X_bool \g enumext minipage right X dim

Internal variables used by enumext* and keyans* environments.

```
\cs_set_protected:Npn \__enumext_tmp:n #1
155
      \bool_new:c { l__enumext_item_starred_#1_bool
156
      \int_new:c { l__enumext_item_column_pos_#1_int }
157
      \int_new:c { g__enumext_item_count_all_#1_int
      \int_new:c { l__enumext_joined_item_#1_int
      \int_new:c { l__enumext_joined_item_aux_#1_int }
      \int_new:c { l__enumext_tmpa_#1_int
      \box_new:c { l__enumext_item_text_#1_box
      \dim_new:c { l__enumext_joined_width_#1_dim
163
      \dim_new:c { l__enumext_item_width_#1_dim
      \tl_new:c { g__enumext_item_symbol_aux_#1_tl
166
      \str_new:c { l__enumext_align_label_#1_str
167
      \bool_new:c { g__enumext_minipage_active_#1_bool }
      \tl_new:c { g__enumext_miniright_code_#1_tl
168
      \bool_new:c { g__enumext_minipage_center_#1_bool }
169
      \dim_new:c { g__enumext_minipage_right_#1_dim
170
      \skip_new:c { g__enumext_minipage_right_#1_skip
\clist_map_inline:nn { vii, viii } { \__enumext_tmp:n {#1} }
```

 $(\textit{End of definition for } \verb|\l_enumext_item_starred_X_bool and others.)$

\c__enumext_all_envs_clist

\g__enumext_minipage_right_X_skip

An internal clist-var variable to run with __enumext_tmp:n.

```
\clist_const:Nn \c__enumext_all_envs_clist
175
      {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv},
      {keyans}{v}, {enumext*}{vii}, {keyans*}{viii}
177
```

(End of definition for $\c_enumert_all_envs_clist$.)

11.5 Some utility functions

__enumext_at_begin_document:n A internal "hook" function used for copying plain list and minipage environments definition and hyperref detection.

```
\cs_new_protected:Npn \__enumext_at_begin_document:n #1
      \hook_gput_code:nnn {begindocument} {enumext} { #1 }
    }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_at_begin_document:n.)$

©2024 by Pablo González L 26 / 125

_enumext_after_env:nn A internal "hook" function for execute code minirigth and minirigth* keys outside the enumext* and keyans* environments and print check-ans outside the enumext and enumext* environments.

```
\cs_new_protected:Npn \__enumext_after_env:nn #1 #2
      \hook_gput_code:nnn {env/#1/after} {enumext} {#2}
    }
```

(End of definition for $_$ enumext_after_env:nn.)

__enumext_level: Function for check current level in enumext.

```
187 \cs_new:Nn \__enumext_level:
188 {
      \int_to_roman:n { \l__enumext_level_int }
189
```

(End of definition for $__$ enumext_level:.)

\ enumext if is int:nT __enumext_if_is_int:nF __enumext_if_is_int:nTF A conditional function to know if the variable we are passing is an integer used by start and widest keys. This function is taken directly from the answer given by Henri Menke in How to test if an expl3 function argument is an integer expression?.

```
prg_new_protected_conditional:Npnn \__enumext_if_is_int:n #1 { T, F, TF }
      \regex_match:nnTF { ^[\+\-]?[\d]+$ } {#1} % $
193
        { \prg_return_true: }
104
        { \prg_return_false: }
196
```

 $(\textit{End of definition for } \\ _\texttt{enumext_if_is_int:nT}, \\ \\ _\texttt{enumext_if_is_int:nF}, \\ \textit{and } \\ \\ \\ _\texttt{enumext_if_is_int:nTF}.)$

_enumext_regex_counter_style:

The internal function __enumext_regex_counter_style: replace the '*' with the actual counter of the running level and is used by the ref key. It loops through the defined counter styles in \c_enumext_counter_style_tl and replace '*' by real command, for example, looking for \arabic* and replacing that by $\langle arabic \langle counter \rangle \rangle$ defined on the current level.

```
\cs_new_protected:Nn \__enumext_regex_counter_style:
     \tl_map_inline:Nn \c__enumext_counter_style_tl
         \regex_replace_once:nnN { \c{##1}\* }
          { \c{\#1}\cB} \cE} \ \l__enumext_ref_key_arg_tl
   }
204
```

(End of definition for __enumext_regex_counter_style:.)

enumext show length:nnn

Internal function used by show-length key to show "all lengths" calculated and use in enumext, enumext*, keyans and keyans* environments.

```
205 \cs_new:Npn \__enumext_show_length:nnn #1 #2 #3
207
      \prg_replicate:nn { 14 - \str_count:n {#2} } { ~ }
208
        = ~ \use:c { #1_use:c } { l__enumext_#2_#3_#1 } \\
209
```

(End of definition for $\label{lem:length:nnn}$)

11.5.1 Utilities for environments and levels

\ enumext is not nested: __enumext_is_on_first_level: The function __enumext_is_not_nested: set the variables \g__enumext_standar_bool and \g__enumext_starred_bool to "true" only if the environments enumext and enumext* are nested in each other.

```
\cs_new_protected:Nn \__enumext_is_not_nested:
    {
      \str_case:en { \@currenvir }
213
        {
214
          {enumext}
216
               \bool_lazy_and:nnT
                 { \bool_not_p:n { \g__enumext_standar_bool } }
                 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
                 {
                   \bool_gset_true:N \g__enumext_standar_bool
```

©2024 by Pablo González L

The function __enumext_is_on_first_level: will set the variables \l__enumext_standar_first_bool and \l__enumext_starred_first_bool to "true" only if the environment is not nested and we are in the "first level" of it . We will also save the start line number of each environment in the variable \g__enumext_start_line_tl and the name of each environment in the variable \g__enumext_envir_name_tl to use in messages related to the check-ans key and .log file.

```
235 \cs_new_protected:Nn \__enumext_is_on_first_level:
    {
236
      \bool_lazy_all:nT
           { \bool_if_p:N \g__enumext_standar_bool }
          { \int_compare_p:nNn { \l__enumext_level_int } = { 1 } }
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
        }
        {
243
           \bool_set_true:N \l__enumext_standar_first_bool
           \tl_gset:Nn \g__enumext_envir_name_tl { enumext }
           \tl_gset:Ne \g__enumext_start_line_tl
               on ~ line ~ \exp_not:V \inputlineno
        }
      \bool_lazy_all:nT
251
        {
           { \bool_if_p:N \g__enumext_starred_bool }
253
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
254
            \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
           \bool_set_true:N \l__enumext_starred_first_bool
           \tl_gset:Nn \g__enumext_envir_name_tl { enumext* }
           \tl_gset:Ne \g__enumext_start_line_tl
               on ~ line ~ \exp_not:V \inputlineno
262
        }
    }
265
```

 $(\textit{End of definition for } \c enumert_is_not_nested: and \c enumert_is_on_first_level:.)$

__enumext_keyans_save_start_line:

The function __enumext_keyans_save_start_line: will save the start line number of the environments keyans, keyans* and keyanspic in the variable \l__enumext _check_start_line_env_tl to use in the __enumext_check_starred_cmd:n function.

```
266 \cs_new_protected:Nn \__enumext_keyans_save_start_line:
    {
267
       \str_case:en { \@currenvir }
268
        {
           {keyans}
             {
               \tl_set:Ne \l__enumext_check_start_line_env_tl
                 {
                   in ~ 'keyans' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
                 }
             }
           {keyans*}
             {
               \tl_set:Ne \l__enumext_check_start_line_env_tl
279
                 {
```

©2024 by Pablo González L

```
in ~ 'keyans*' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
}

keyanspic}

{
keyanspic}

{
tl_set:Ne \l__enumext_check_start_line_env_tl

in ~ 'keyanspic' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
}

in ~ 'keyanspic' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
}

}
```

 $(\textit{End of definition for } \verb|_-enumext_keyans_save_start_line:.)$

11.5.2 Utilities for log and terminal

The function __enumext_reset_global_vars: will be passed to the function __enumext_execute_-after_env: and will return the global variables to their default values after being used.

```
_{^{293}} \cs_new_protected:Nn \__enumext_reset_global_vars:
       \__enumext_reset_global_int:
       \__enumext_reset_global_bool:
       \__enumext_reset_global_tl:
298
    }
299 \cs_new_protected:Nn \__enumext_reset_global_int:
300
       \int_gzero:N \g__enumext_item_number_int
301
       \int_gzero:N \g__enumext_item_anskey_int
302
       \int_gzero:N \g__enumext_item_answer_diff_int
303
304
305 \cs_new_protected:Nn \__enumext_reset_global_bool:
       \bool_gset_false:N \g__enumext_standar_bool
       \bool_gset_false:N \g__enumext_starred_bool
    }
310
\cs_new_protected:Nn \__enumext_reset_global_tl:
       \tl_gclear:N \g__enumext_store_name_tl
      \tl_gclear:N \g__enumext_start_line_tl
       \tl_gclear:N \g__enumext_envir_name_tl
(End of definition for \_enumext_reset_global_vars: and others.)
```

__enumext_log_global_vars:
__enumext_log_answer_vars:

__enumext_reset_global_vars:

_enumext_reset_global_tl:

The function __enumext_log_global_vars: will be passed to the function __enumext_execute_-after_env: and write to the .log file the number of elements saved in the $\langle prop \; list \rangle$ and $\langle sequence \rangle$ created by the save-ans key along with the value of the integer variable created for the resume key.

The function __enumext_log_answer_vars: will be passed to the function __enumext_execute_-after_env: and write to the .log file the number of items and answers along with the difference between them.

 $(\textit{End of definition for } \c\c enumert_log_global_vars: and \c\c enumert_log_answer_vars:.)$

©2024 by Pablo González L 29/125

11.6 Copying list and minipage environments

The list environment provided by LTFX has the following plain form:

```
\label{eq:cont} $$ \left( \left\langle arg \ one \right\rangle \right) \left( \left\langle arg \ two \right\rangle \right) $$ \left(
```

As a precaution we copy them using __enumext_at_begin_document:n in case any package redefines the list environment or a related command.

__enumext_start_list:nn
 __enumext_stop_list:
 __enumext_item_std:w

The functions __enumext_start_list:nn, __enumext_stop_list: and __enumext_item_-std:w correspond to copies of \list, \endlist and \item from plain definition of list environment.

```
332 \__enumext_at_begin_document:n
333 {
334     \cs_new_eq:NN \__enumext_start_list:nn \list
335     \cs_new_eq:NN \__enumext_stop_list: \endlist
336     \cs_new_eq:NN \__enumext_item_std:w \item
337 }
```

(End of definition for __enumext_start_list:nn, __enumext_stop_list:, and __enumext_item_std:w.)

The minipage environment provided by LaTeX has the following (simplified) plain form:

```
\begin{tabular}{ll} $$ \min[age[\langle pos \rangle][\langle height \rangle][\langle inner-pos \rangle]\{\langle width \rangle\} \\ & \langle internal\ implement \rangle \\ \end{tabular}
```

As a precaution we copy them using __enumext_at_begin_document:n in case any package redefines the minipage environment or a related command.

__enumext_minipage:w
__enumext_endminipage:

The functions __enumext_minipage:w, __enumext_endminipage: and correspond to copies of \minipage, \endminipage from plain definition of minipage environment.

```
338 \__enumext_at_begin_document:n
339 {
340      \cs_new_eq:NN \__enumext_minipage:w \minipage
341      \cs_new_eq:NN \__enumext_endminipage: \endminipage
342 }
```

(End of definition for __enumext_minipage:w and __enumext_endminipage:.)

11.7 Compatibility with hyperref and footnotehyper

First we define the necessary rules using "hooks" to determine if the hyperref package is loaded.

```
_{^{343}} \hook_gput_code:nnn { begindocument } { enumext } { \__enumext_after_hyperref: } _{^{344}} \hook_gset_rule:nnnn { begindocument } { enumext } { after } { hyperref }
```

__enumext_after_hyperref:
__enumext_hypertarget:nn
__enumext_phantomsection:

The function __enumext_after_hyperref: sets the state of the boolean variable \l__enumext_-hyperref_bool to "true" if the package is loaded. At this point we will use the public macro \IfHyperBoolean to determine if the hyperfootnotes=true key is present, if so, we set the state of the boolean variable __enumext_footnotes_key_bool to "true".

If the state of the variable \l__enumext_footnotes_key_bool is true we will check if the package footnotehyper is loaded, in case it is not present, we will set the value of \l__enumext_footnotes_-key_bool to false and we will redefine \footnote.

```
359 \bool_if:NT \l__enumext_footnotes_key_bool
360 {
361 \IfPackageLoadedTF { footnotehyper }
362 {
```

©2024 by Pablo González L

```
\msg_info:nnn { enumext } { package-load } { footnotehyper }

\msg_info:nnn { enumext } { package-load } { footnotehyper }

\msg_info:nnn { enumext } { package-load } {

\msg_info:nnn { enumext } { \msg_info:nnn { enumext } { } {

\msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnnn { enumext } { \msg_info:nnn { enumext } { \msg_info:nnn { enum
```

The functions __enumext_hypertarget:nn and __enumext_phantomsection: correspond to the internal copies of \hypertarget and \phantomsection. If the boolean variable \l__enumext_-hyperref_bool is false the functions __enumext_hypertarget:nn and __enumext_phantomsection: will be disabled.

 $(\textit{End of definition for } \verb|_= numext_after_hyperref: , \verb|_= numext_hypertarget:nn, and \verb|_= numext_phantomsection:.)$

__enumext_newlabel:nn

The function __enumext_newlabel:nn write the information to the .aux file when using the save-ref key. The arguments taken by the function are:

```
#1: \l_enumext_newlabel_arg_one_tl
#2: \l_enumext_newlabel_arg_two_tl
```

The trick here is to manage the number of arguments passed to \newlabel{#1}{#2} according to the presence of the hyperref package.

```
381 \cs_new_protected:Npn \__enumext_newlabel:nn #1 #2
382
       \protected@write \@auxout { }
383
384
           \token_to_str:N \newlabel {#1}
             {
               {#2}
               \bool_if:NT \l__enumext_hyperref_bool
                 { { \thepage } {#2} {#1} }
               { }
             }
391
         }
392
       \__enumext_hypertarget:nn {#1} { }
393
       \__enumext_phantomsection:
394
395
```

 $(End\ of\ definition\ for\ \verb|_-enumext_newlabel:nn.|)$

11.8 Definition of counters

__enumext_define_counters:Nn \ enumext_define_counters:cn To create the necessary "counters" we must first make sure that they are not already defined by the user or a package such as enumitem, otherwise a error will be returned and the package loading will be aborted. The arguments taken by the function are:

31/125

#1: A token list \l__enumext_counter_X_tl for "store" the counter's name.

#2: The counter's name.

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_define_counters:Nn.|)$

©2024 by Pablo González L

The counters created here are enumXi, enumXii, enumXiii and enumXiv for enumext environment, enumXi enumXii enumXv for keyans environment, enumXvi for keyanspic environment, enumXvii for enumext* and enumXviii for the keyans* environments. enumXiii

```
enumXiv
           405 \__enumext_define_counters:Nn \l__enumext_counter_i_tl { enumXi
  enumXv
          406 \__enumext_define_counters:Nn \l__enumext_counter_ii_tl { enumXii
 \verb"enumXvi" 407 $$ \enumext_define_counters: Nn \l_enumext_counter_iii_tl $ \{ enumXiii \} $ \} $$
enumXvii 408 \__enumext_define_counters:Nn \l__enumext_counter_iv_tl { enumXiv
enumXviii 49 \__enumext_define_counters:Nn \l__enumext_counter_v_tl { enumXv
           410 \__enumext_define_counters:Nn \l__enumext_counter_vi_tl { enumXvi
           411 \__enumext_define_counters:Nn \l__enumext_counter_vii_tl { enumXvii }
           412 \__enumext_define_counters:Nn \l__enumext_counter_viii_tl { enumXviii }
```

(End of definition for enumXi and others.)

11.9 Definition of labels

This part of the code is inspired by the enumitem package. The idea is to be able to access the counters using \arabic*, \Alph*, \alph*, \Roman* and \roman* to use them in the label key.

__enumext_register_counter_style:Nn

These (counters) will be used as default (labels) if the label key is not used for the different levels of the enumext environment and the keyans environment, so it is necessary to get a default value for labelwidth from these (*labels*) at the same time.

```
413 \cs_new_protected:Npn \__enumext_register_counter_style:Nn #1 #2
414 {
      \tl_const:cn { c__enumext_widest_ \cs_to_str:N #1 _tl } {#2}
415
      \tl_gput_right:Nn \g__enumext_counter_styles_tl {#1}
416
417
418 \__enumext_register_counter_style:Nn \arabic { 0 }
419 \__enumext_register_counter_style:Nn \Alph { M }
420 \__enumext_register_counter_style:Nn \alph { m }
421 \__enumext_register_counter_style:Nn \Roman { VIII }
_{422} \__enumext_register_counter_style:Nn \roman { viii }
```

(End of definition for __enumext_register_counter_style:Nn.)

__enumext_label_width_by_box:cv

no labelwidth key is passed.

```
423 \cs_new_protected:Npn \__enumext_label_width_by_box:Nn #1 #2
      \hbox_set:Nn \l__enumext_label_width_by_box {#2}
      \dim_set:Nn #1 { \box_wd:N \l__enumext_label_width_by_box }
    }
_{\mbox{\tiny 428}} \cs_generate_variant:Nn \__enumext_label_width_by_box:Nn { cv }
```

(End of definition for $\label{lem:label_width_by_box:Nn.}$)

\ enumext label style:Nnn __enumext_label_style:cvn The function __enumext_label_style: Nnn is used by the label key to creates the variables containing the \(\lambda label style\) and will allow to use \arabic*, \Alph*, \alph*, \Roman* and \roman* as arguments. It loops through the defined counter styles in \g__enumext_counter_styles_tl (\arabic, \alph, Alph, \roman, and \Roman) for example, looking for \roman* and replacing that by \roman{\current} counter\}, and doing the same for the $\g_{\text{enumext_widest_label_tl}}$ to keep both in sync.

```
429 \cs_new_protected:Npn \__enumext_label_style:Nnn #1 #2 #3
430
      \tl_clear_new:N #1
431
      \tl_put_right:Ne #1 { \tl_trim_spaces:n {#3} }
432
      \tl_gset_eq:NN \g__enumext_widest_label_tl #1
433
      \tl_map_inline:Nn \g__enumext_counter_styles_tl
434
        {
435
          \tl_replace_all:Nne #1 { ##1* } { \exp_not:N ##1 {#2} }
          \tl_greplace_all:Nne \g__enumext_widest_label_tl { ##1* }
            { \tl_use:c { c__enumext_widest_ \cs_to_str:N ##1 _tl } }
      \__enumext_label_width_by_box:Nn \l__enumext_current_widest_dim
        { \tl_use:N \g__enumext_widest_label_tl }
      \tl_set_eq:cN { the #2 } #1
442
443
444 \cs_generate_variant:Nn \__enumext_label_style:Nnn { cvn }
```

©2024 by Pablo González L

(End of definition for $_$ enumext_label_style:Nnn.)

11.10 Setting keys associated with label

Definition of keys font, labelsep, labelwidth, wrap-label and wrap-label* keys for enumext and keyans environments. labelsep labelwidth 445 \cs_set_protected:Npn __enumext_tmp:nn #1 #2 wrap-label 446 wrap-label* \keys_define:nn { enumext / #1 } 447 { font .tl_set:c = { l__enumext_label_font_style_#2_tl }, font .value_required:n = true, labelsep .dim_set:c = { l__enumext_labelsep_#2_dim }, labelsep .initial:n = $\{0.3333em\}$, 452 labelsep .value_required:n = true, 453 labelwidth .dim_set:c = { l__enumext_labelwidth_#2_dim }, 454 labelwidth .value_required:n = true, 455 wrap-label .cs_set_protected:cp = { __enumext_wrapper_label_#2:n } ##1, 456 wrap-label .initial:n = {##1}, 457 wrap-label .value_required:n = true, 458 wrap-label* .code:n = { 459 \bool_set_true:c { l__enumext_wrap_label_opt_#2_bool } \keys_set:nn { enumext / #1 } { wrap-label = {##1} } }, wrap-label* .value_required:n = true, } 464 466 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

(End of definition for font and others.)

In this point, the following are set __enumext_wrapper_label_X:n which will be used by __enumext_make_-label: for the different levels of the enumext environment and is set to __enumext_wrapper_label_v:n which will be used by __enumext_keyans_make_label: for keyans and keyanspic environments.

align The align key is implemented differently for "starred" and "non starred" environments.

```
467 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
     {
468
       \keys_define:nn { enumext / #1 }
469
         {
           align .choice:,
471
           align / left
                          .code:n =
472
                             {
473
                               \tl_clear:c { l__enumext_label_fill_left_#2_tl }
474
                               \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill }
475
                             },
           align / right .code:n =
                               \tl_set:cn { l__enumext_label_fill_left_#2_tl } { \hfill }
                               \tl_clear:c { l__enumext_label_fill_right_#2_tl }
                             },
           align / center .code:n =
                               \tl_set:cn { l__enumext_label_fill_left_#2_tl } { \hfill }
                               \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill }
                              },
           align .initial:n = left,
           align .value_required:n = true,
         }
491 \clist_map_inline:nn
    {
492
       {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {keyans}{v}
493
494
     { \__enumext_tmp:nn #1 }
495
496 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
       \keys_define:nn { enumext / #1 }
         {
           align .choice:,
           align / left
                          .code:n = \str_set:cn { l__enumext_align_label_#2_str } { l },
           align / right .code:n = \str_set:cn { l__enumext_align_label_#2_str } { r },
           align / center .code:n = \str_set:cn { l__enumext_align_label_#2_str } { c },
©2024 by Pablo González L
```

11.11 Setting label and ref keys

(End of definition for align.)

The implementation of the keys label and ref are part of the core of the package enumext, here the default values for $\langle label \rangle$, the value of the variables $\l_enumext_label_X_tl$, the default values for $\l_enumext_label_X_tl$, and $\l_enumext_label_X_tl$, the default values for $\l_enumext_label_X_tl$, the default values for $\l_enumext_label_X_tl$, and $\l_enumext_label_X_tl$, the default values for $\l_enumext_label_X_tl$, and $\l_enumext_label_X$

11.11.1 Define and set label and ref keys for enumext environment

label Here we set the default $\langle labels \rangle$ of the *four levels* of enumext environment, along with the default value for ref labelwidth key and ref key.

```
\l__enumext_label_i_tl
\l__enumext_label_ii_tl
\l__enumext_label_iii_tl
\l__enumext_label_iv_tl
```

```
509 \cs_set_protected:Npn \__enumext_tmp:nnn #1 #2 #3
      \keys_define:nn { enumext / #1 }
       {
         label .code:n
                             \__enumext_label_style:cvn { l__enumext_label_#2_tl }
                               { l__enumext_counter_#2_tl } {##1}
                             \dim_set_eq:cN { l__enumext_labelwidth_#2_dim }
516
                               \l__enumext_current_widest_dim
                           1.
518
         label .initial:n = #3,
         label .value_required:n = true,
         ref
               .code:n = \__enumext_standar_ref:n {##1},
               .value_required:n = true,
         ref
       }
    }
  526 \__enumext_tmp:nnn { level-2 } { ii } { (\alph*) }
527 \__enumext_tmp:nnn { level-3 } { iii } { \roman*. }
528 \__enumext_tmp:nnn { level-4 } { iv } { \Alph*. }
```

(End of definition for label and others.)

__enumext_standar_ref:n
__enumext_standar_ref:

The __enumext_standar_ref:n first we will pass the key argument to \l__enumext_ref_key_arg_tl and we will analyze its state, if it is not <code>empty</code> we will make a copy of the current counter in \l__enumext _ref_the_count_tl and we will execute the function __enumext_regex_counter_style: which will return the modified \l__enumext_ref_key_arg_tl and we make the value of \l__enumext_ref_the_count_tl the same as that \l__enumext_the_counter_X_tl which contains \theenumX and finally we set \l__enumext_renew_the_count_X_tl with the renewed command.

```
_{529} \cs_new_protected:Npn \__enumext_standar_ref:n #1
530
      \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
531
      \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
532
        {
           \msg_error:nnn { enumext } { key-ref-empty } { enumext }
534
        }
535
         {
536
           \tl_set_eq:Nc
             \l__enumext_ref_the_count_tl { l__enumext_counter_ \__enumext_level: _tl }
             _enumext_regex_counter_style:
           \tl_set_eq:Nc
             \l__enumext_ref_the_count_tl { l__enumext_the_counter_ \__enumext_level: _tl }
           \tl_put_right:ce { l__enumext_renew_the_count_ \__enumext_level: _tl }
               \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
                 { \exp_not:V \l__enumext_ref_key_arg_tl }
        }
547
```

Finally the function __enumext_standar_ref: will execute the modification for the reference system in the second argument of the environment definition enumext.

```
549 \cs_new_protected:Nn \__enumext_standar_ref:
550 {

©2024 by Pablo González L
```

```
\tl_if_empty:cF { l__enumext_renew_the_count_ \__enumext_level: _tl }
    \tl_use:c { l__enumext_renew_the_count_ \__enumext_level: _tl }
  }
```

(End of definition for $\ \ \$ enumext_standar_ref:n and $\ \ \ \ \$ enumext_standar_ref:.)

11.11.2 Define and set label and ref keys for enumext* and keyans* environments

Here we set the default $\langle labels \rangle$ for enumext* and keyans* environments, along with the default value for labelwidth key and ref key.

\l__enumext_label_vii_tl \l__enumext_label_viii_tl

```
556 \cs_set_protected:Npn \__enumext_tmp:nnn #1 #2 #3
    {
557
       \keys_define:nn { enumext / #1 }
558
559
          label .code:n
                            = {
                                 \__enumext_label_style:cvn { l__enumext_label_#2_tl }
                                   { l__enumext_counter_#2_tl } {##1}
                                \dim_set_eq:cN { l__enumext_labelwidth_#2_dim }
                                  \l__enumext_current_widest_dim
          label .initial:n = #3,
566
          label .value_required:n = true,
          ref
                 .code:n
                            = \__enumext_starred_ref:n {##1},
          ref
                 .value_required:n = true,
        }
    }
572 \__enumext_tmp:nnn { enumext* } { vii } { \arabic*.}
573 \__enumext_tmp:nnn { keyans* } { viii } { (\Alph*) }
```

(End of definition for label and others.)

__enumext_starred_ref:n __enumext_starred_ref:

The implementation of __enumext_starred_ref:n is the same as that used for the environment enumext.

```
574 \cs_new_protected:Npn \__enumext_starred_ref:n #1
    {
      \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
576
      \int_compare:nNnT { \l__enumext_level_h_int } = { 1 }
           \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
            {
               \msg_error:nnn { enumext } { key-ref-empty } { enumext* }
581
            7
582
583
            {
               \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_vii_tl
584
               \__enumext_regex_counter_style:
585
               \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_the_counter_vii_tl
               \tl_put_right:Ne \l__enumext_renew_the_count_vii_tl
                 {
                   \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
                     { \exp_not:V \l__enumext_ref_key_arg_tl }
                 }
            }
593
      \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
594
        {
           \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
596
            {
               \msg_error:nnn { enumext } { key-ref-empty } { keyans* }
            }
             {
              \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_viii_tl
               \__enumext_regex_counter_style:
               \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_the_counter_viii_tl
603
               \tl_put_right:Ne \l__enumext_renew_the_count_viii_tl
                   \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
                     { \exp_not:V \l__enumext_ref_key_arg_tl }
                 }
            }
        }
611
```

©2024 by Pablo González L

Finally the function __enumext_starred_ref: will execute the modification for the reference system in the second argument of the enumext* and keyans* environment definition.

 $(\mathit{End}\ of\ definition\ for\ \verb|_enumext_starred_ref:n\ and\ \verb|_enumext_starred_ref:|)$

11.11.3 Define and set label and ref keys for keyans and keyanspic environments

Here we set the default $\langle label \rangle$ for keyans and keyanspic environment, along with the default value for labelwidth and ref key. The keyanspic environment use the same $\langle label \rangle$ as the keyans environment.

```
\l_enumext_label_v_tl
\l_enumext_label_vi_tl
```

label

```
629 \keys_define:nn { enumext / keyans }
630
       label .code:n
631
                             \__enumext_label_style:cvn { l__enumext_label_v_tl }
632
                               { l__enumext_counter_v_tl } {#1}
633
                             \dim_set_eq:cN { l__enumext_labelwidth_v_dim }
634
                              \l__enumext_current_widest_dim
                             \__enumext_label_style:cvn { l__enumext_label_vi_tl }
                                { l__enumext_counter_vi_tl } {#1}
                             \dim_set_eq:cN { l__enumext_labelwidth_v_dim }
                                \l__enumext_current_widest_dim
                          }.
      label .initial:n = (\Alph*),
641
      label .value_required:n = true,
642
      ref
             .code:n
                        = \__enumext_keyans_ref:n {#1},
643
      ref
             .value_required:n = true,
644
645
```

(End of definition for label and others.)

__enumext_keyans_ref:n
__enumext_keyans_ref:

The implementation of __enumext_keyans_ref:n is the same as that used for the environment enumext.

```
646 \cs_new_protected:Npn \__enumext_keyans_ref:n #1
647
      \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
648
      \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
649
        {
650
           \msg_error:nnn { enumext } { key-ref-empty } { keyans }
651
        }
652
653
          \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_v_tl
           \__enumext_regex_counter_style:
           \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_the_counter_v_tl
           \tl_put_right:Ne \l__enumext_renew_the_count_v_tl
            {
658
               \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
                 { \exp_not:V \l__enumext_ref_key_arg_tl }
        }
```

Finally the function __enumext_keyans_ref: will execute the modification for the reference system in the second argument of the keyans* environment definition.

```
664 \cs_new_protected:Nn \__enumext_keyans_ref:
665 {
666 \tl_if_empty:NF \l__enumext_renew_the_count_v_tl
©2024 by Pablo González L
```

(End of definition for $_$ enumext_keyans_ref:n and $_$ enumext_keyans_ref:.)

11.12 Setting start and widest keys

__enumext_start_from:NNn
_enumext_start_from:ccn

The function $__$ enumext_start_from:NNn used by the start key take three arguments:

```
#1: \l__enumext_label_X_tl
#2: \l__enumext_start_X_int
#3: \langle integer or string \rangle
```

The first argument of this function are the "counter style" set by label key, the second argument is returned by the function, the third argument can be an $\langle integer \rangle$ or $\langle string \rangle$ of the form $\arrowvert Alph$, \ar

```
671 \cs_new_protected:Npn \__enumext_start_from:NNn #1 #2 #3
    {
672
      \__enumext_if_is_int:nTF { #3 }
673
            \int_set:Nn #2 {#3}
675
         }
          {
            \regex_match:nVT { \c{Alph} | \c{alph} } {#1}
              { \int_set:Nn #2 { \int_from_alph:n {#3} } }
            \regex_match:nVT { \c{Roman} | \c{roman} } {#1}
              { \int_set:Nn #2 { \int_from_roman:n {#3} } }
681
         }
682
684 \cs_generate_variant:Nn \__enumext_start_from:NNn { ccn }
```

 $(\textit{End of definition for } \verb|_-enumext_start_from: NNn.)$

__enumext_widest_from:nNNn
__enumext_widest_from:nccn

The function __enumext_widest_from: nNNn used by the widest key take four arguments:

#1: The counter associated with the environment level

```
#2: \l_enumext_label_X_tl
```

#3: \l_enumext_labelwidth_X_dim
#4: \langle integer or string \rangle

The second and third arguments of this function are the values set by label and labelwidth keys, the four argument can be an $\langle integer \rangle$ or $\langle string \rangle$ of the form \Alph, \alph, \Roman or \roman. The value of the four argument is set temporarily for the identified counter in this point (level), then the value is expanded into a "box" and the "width" of the "box" is returned.

```
\cs_new_protected:Npn \__enumext_widest_from:nNNn #1 #2 #3 #4
         _enumext_if_is_int:nTF {#4}
        {
          \setcounter{enumX#1} { #4 }
        }
        {
          \regex_match:nVT { \c{Alph} | \c{alph} } {#2}
602
             { \setcounter{enumX#1} { \int_from_alph:n {#4} } }
693
          \regex_match:nVT { \c{Roman} | \c{roman} } {#2}
694
             { \setcounter{enumX#1} { \int_from_roman:n {#4} } }
695
        \__enumext_label_width_by_box:cv
         { l__enumext_labelwidth_#1_dim } { l__enumext_label_#1_tl }
698
\cs_generate_variant:Nn \__enumext_widest_from:nNNn { nccn }
```

(End of definition for $_$ enumext_widest_from:nNNn.)

tart Now define and set start and widest keys for enumext and keyans environments.

```
start
widest
\l__enumext_start_X_int
```

parsep

nosep

noitemsep

($End\ of\ definition\ for\ start$, widest, and \l_enumext_start_X_int.)

11.13 Setting keys for vertical spaces

topsep Define and set topsep, partopsep, parsep, itemsep, noitemsep and nosep keys for enumext and partopsep keyans environments.

```
721 \cs_set_protected:Npn \__enumext_tmp:nnnnnn #1 #2 #3 #4 #5 #6
722
      \keys_define:nn { enumext / #1 }
724
        {
          topsep
                    .skip_set:c = { l__enumext_topsep_#2_skip },
725
          topsep
                    .initial:n = \{#3\},
726
          topsep
                    .value_required:n = true,
          partopsep .skip_set:c = { l__enumext_partopsep_#2_skip },
728
          partopsep .initial:n = {#4},
          partopsep .value_required:n = true,
          parsep
                  .skip_set:c = { l__enumext_parsep_#2_skip },
                    .initial:n = {#5},
          parsep
                     .value_required:n = true,
          parsep
733
                    .skip_set:c = { l__enumext_itemsep_#2_skip },
          itemsep
734
          itemsep
                     .initial:n = \{\#6\},
735
          itemsep
                     .value_required:n = true,
736
          noitemsep .meta:n
                              = { itemsep = 0pt, parsep = 0pt },
737
          noitemsep .value_forbidden:n = true,
738
          nosep
                     .meta:n
                                = {
                                     itemsep = 0pt, parsep= 0pt,
                                     topsep = Opt, partopsep = Opt,
                                   },
                     .value_forbidden:n = true,
          nosep
        }
```

Now we set the values based on standard article class in 10pt.

```
746 \__enumext_tmp:nnnnnn { level-1 } { i } { 8.0pt plus 2.0pt minus 4.0pt }
   { 2.0pt plus 1.0pt minus 1.0pt } { 4.0pt plus 2.0pt minus 1.0pt }
    { 4.0pt plus 2.0pt minus 1.0pt }
749 \__enumext_tmp:nnnnnn { level-2 } { ii } { 4.0pt plus 2.0pt minus 1.0pt }
    { 2.0pt plus 1.0pt minus 1.0pt } { 2.0pt plus 1.0pt minus 1.0pt }
    { 2.0pt plus 1.0pt minus 1.0pt }
752 \__enumext_tmp:nnnnnn { level-3 } { iii } { 2.0pt plus 1.0pt minus 1.0pt }
   { 1.0pt minus 1.0pt }{ 0pt }{ 2.0pt plus 1.0pt minus 1.0pt }
754 \__enumext_tmp:nnnnnn { level-4 } { iv } { 2.0pt plus 1.0pt minus 1.0pt }
  { 1.0pt minus 1.0pt }{ 0pt }{ 2.0pt plus 1.0pt minus 1.0pt }
756 \__enumext_tmp:nnnnnn { keyans } { v }{ 4.0pt plus 2.0pt minus 1.0pt }
  { 2.0pt plus 1.0pt minus 1.0pt }{ 2.0pt plus 1.0pt minus 1.0pt }
  { 2.0pt plus 1.0pt minus 1.0pt }
759 \__enumext_tmp:nnnnnn { enumext* } { vii } { 8.0pt plus 2.0pt minus 4.0pt }
760 { 2.0pt plus 1.0pt minus 1.0pt } { 4.0pt plus 2.0pt minus 1.0pt }
    { 4.0pt plus 2.0pt minus 1.0pt }
_{762} \__enumext_tmp:nnnnnn { keyans* } { viii } { 4.0pt plus 2.0pt minus 1.0pt }
763 { 2.0pt plus 1.0pt minus 1.0pt } { 2.0pt plus 1.0pt minus 1.0pt }
    { 2.0pt plus 1.0pt minus 1.0pt }
```

(End of definition for topsep and others.)

11.14 Setting keys for horizontal spaces

rightmargin enumext and keyans environments.

listparindent list-offset ©2024 by Pablo González L

itemindent Define and set itemindent, rightmargin, listparindent, list-offset and list-indent keys for enumext and keyans environments.

```
765 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
      \keys_define:nn { enumext / #1 }
        {
                        .dim_set:c = { l__enumext_fake_item_indent_#2_dim },
          itemindent
          itemindent
                        .value_required:n = true,
          rightmargin .dim_set:c = { l__enumext_rightmargin_#2_dim },
          rightmargin .value_required:n = true,
          listparindent .dim_set:c = { l__enumext_listparindent_#2_dim },
          listparindent .value_required:n = true,
          list-offset
                        .dim_set:c = { l__enumext_listoffset_#2_dim },
          list-offset
                        .value_required:n = true,
          list-indent
                        .code:n
                          \bool_set_true:c { l__enumext_leftmargin_tmp_#2_bool }
778
                          \dim_set:cn { l__enumext_leftmargin_tmp_#2_dim } {##1},
          list-indent
                        .value_required:n = true,
781
782
783 \clist_map_inline:Nn \c__enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

(End of definition for itemindent and others.)

©2024 by Pablo González L

For enumext* and keyans* environments the situation is a bit different, the list-indent key behaves like the list-offset key.

11.14.1 Functions for setting the fake itemindent

The itemindent key does not set the value of \itemindent, it only sets the value of the *horizontal space* applied using \skip_horizontal:N. We will store this value in the variable and only apply it when it is greater than <code>Opt</code>. Here I will need to place \mode_leave_vertical: and the plain TeX macro \ignorespaces to avoid unwanted extra space when using the itemindent key.

```
789 \cs_set_protected:Nn \__enumext_fake_item:
      \dim_compare:nNnT
        { \dim_use:c { l__enumext_fake_item_indent_ \__enumext_level: _dim } }
        { \c_zero_dim }
          \tl_set:ce { l__enumext_fake_item_indent_ \__enumext_level: _tl }
               \exp_not:N \mode_leave_vertical:
               \exp_not:n { \skip_horizontal:n }
                 { \dim_use:c { l__enumext_fake_item_indent_ \__enumext_level: _dim } }
               \ignorespaces
        }
     }
805 \cs_set_protected:Nn \__enumext_keyans_fake_item:
806
      \dim_compare:nNnT
807
        { \l__enumext_fake_item_indent_v_dim } > { \c_zero_dim }
808
        {
           \tl_set:Ne \l__enumext_fake_item_indent_v_tl
               \exp_not:N \mode_leave_vertical:
               \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_v_dim
814
815
816
817 \cs_set_protected:Nn \__enumext_fake_item_vii:
818
      \dim compare:nNnT
819
        { \l__enumext_fake_item_indent_vii_dim } > { \c_zero_dim }
820
           \tl_set:Ne \l__enumext_fake_item_indent_vii_tl
               \exp_not:N \mode_leave_vertical:
```

39 / 125

__enumext_fake_item:
__enumext_keyans_fake_item:
__enumext_fake_item_vii:
__enumext_fake_item_viii:

```
\exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_vii_dim
826
        }
827
     }
828
829 \cs_set_protected:Nn \__enumext_fake_item_viii:
830
       \dim compare:nNnT
831
         { \l__enumext_fake_item_indent_viii_dim } > { \c_zero_dim }
832
833
           \tl_set:Ne \l__enumext_fake_item_indent_viii_tl
               \exp_not:N \mode_leave_vertical:
               \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_viii_dim
837
828
         }
839
840
```

(End of definition for __enumext_fake_item: and others.)

11.15 Setting show-length key

show-length

before

Define and set show-length key for enumext, enumext*, keyans and keyans* environments. The function sets the boolean variable \l__enumext_show_length_X_bool used in the definition of all environments to "true" and calls the function __enumext_show_length:nnn which prints all the values of the "vertical" and "horizontal" parameters calculated and used.

(End of definition for show-length.)

11.16 Setting before, after and first keys

Define and set before, before*, after and first keys for enumext and keyans environments.

```
before*
         850 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
 after
  first
                \keys_define:nn { enumext / #1 }
         853
                    before .tl_set:c = { l__enumext_before_no_starred_key_#2_tl },
         854
                    before .value_required:n = true,
                    before* .tl_set:c = { l__enumext_before_starred_key_#2_tl },
         856
                    before* .value_required:n = true,
         857
                    after
                            .tl_set:c = { l__enumext_after_stop_list_#2_tl },
         858
                    after
                            .value_required:n = true,
         859
                    first
                            .tl_set:c = { l__enumext_after_list_args_#2_tl },
                    first
                            .value_required:n = true,
         864 \clist_map_inline:Nn \c__enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

(End of definition for before and others.)

11.16.1 Functions for before, after and first keys in enumext

__enumext_before_args_exec:
__enumext_before_keys_exec:
_enumext_after_stop_list:
_enumext_after_args_exec:

The function __enumext_before_args_exec: executes the $\{\langle code \rangle\}$ set by the before* key "before" the enumext environment is started. The $\{\langle code \rangle\}$ is executed "without" knowing any definition of the second argument of the list.

```
865 \cs_new_protected:Nn \__enumext_before_args_exec:
866 {
867 \tl_use:c { l__enumext_before_starred_key_ \__enumext_level: _tl }
868 }
```

The function __enumext_before_keys_exec: executes the $\{\langle code \rangle\}$ set by the before key "before" the enumext environment is started in second argument of the list. The $\{\langle code \rangle\}$ is executed "knowing" all definition and values provides by $\langle keys \rangle$.

```
869 \cs_new_protected:Nn \__enumext_before_keys_exec:
870 {
©2024 by Pablo González L
```

```
873 \cs_new_protected:Nn \__enumext_after_stop_list:
874 {
875 \tl_use:c { l__enumext_after_stop_list_ \__enumext_level: _tl }
876 }
```

The function __enumext_after_args_exec: executes the $\{\langle code \rangle\}$ set by the first key after the end of the second argument of the list defining the enumext environment, just before the first occurrence of \item.

```
s<sub>77</sub> \cs_new_protected:Nn \__enumext_after_args_exec:
s<sub>78</sub> {
s<sub>79</sub>   \tl_use:c { l__enumext_after_list_args_ \__enumext_level: _tl }
s<sub>80</sub> }
```

 $(\textit{End of definition for } \verb|_-enumext_before_args_exec: and others.)$

11.16.2 Functions for before, after and first keys in keyans

_enumext_before_args_exec_v: The function __enumext_before_keys_exec_v: the keyans environm _enumext_after_stop_list_v: $\{\langle arg\ two \rangle\}$ of the list.

The function __enumext_before_args_exec_v: executes the $\{\langle code \rangle\}$ set by the before* key "before" the keyans environment is started. The $\{\langle code \rangle\}$ is executed "without" knowing any definition of the $\{\langle arg\ two \rangle\}$ of the list.

```
881 \cs_new_protected:Nn \__enumext_before_args_exec_v:
882 {
883 \tl_use:N \l__enumext_before_starred_key_v_tl
884 }
```

The function __enumext_before_keys_exec_v: executes the $\{\langle code \rangle\}$ set by the before key "before" the keyans environment is started in $\{\langle arg\ two \rangle\}$ of the list. The $\{\langle code \rangle\}$ is executed "knowing" all definition and values provides by $\langle keys \rangle$.

```
885 \cs_new_protected:Nn \__enumext_before_keys_exec_v:
886 {
887 \tl_use:N \l__enumext_before_no_starred_key_v_tl
888 }
```

The function __enumext_after_stop_list_v: executes the $\{\langle code \rangle\}$ set by the after key "after" the keyans environment has finished.

```
889 \cs_new_protected:Nn \__enumext_after_stop_list_v:
890 {
891 \tl_use:N \l__enumext_after_stop_list_v_tl
892 }
```

The function __enumext_after_args_exec_v: executes the $\{\langle code \rangle\}$ set by the first key after the end of $\{\langle arg\ two \rangle\}$ of the list defining the keyans environment, just before the first occurrence of \item.

```
893 \cs_new_protected:Nn \__enumext_after_args_exec_v:
894 {
895 \tl_use:N \l__enumext_after_list_args_v_tl
896 }
```

 $(\textit{End of definition for } \verb|_-enumext_before_args_exec_v: and others.)$

11.16.3 Functions for before, after and first keys in enumext* and keyans*

__enumext_before_args_exec_vii:
__enumext_before_keys_exec_vii:
__enumext_after_stop_list_vii:
__enumext_after_args_exec_vii:

The function __enumext_before_args_exec_v: executes the $\{\langle code \rangle\}$ set by the before* key "before" the keyans environment is started. The $\{\langle code \rangle\}$ is executed "without" knowing any definition of the $\{\langle arg\ two \rangle\}$ of the list.

```
897 \cs_new_protected:Nn \__enumext_before_args_exec_vii:
898 {
899    \tl_use:N \l__enumext_before_starred_key_vii_tl
900 }
901 \cs_new_protected:Nn \__enumext_before_args_exec_viii:
902    {
903    \tl_use:N \l__enumext_before_starred_key_viii_tl
904 }
```

The functions __enumext_before_keys_exec_vii: and __enumext_before_keys_exec_viii: executes the $\{\langle code \rangle\}$ set by the before key "before" in enumext* and keyans* environments is started in $\{\langle arg\ two \rangle\}$ of the list. The $\{\langle code \rangle\}$ is executed "knowing" all definition and values provides by $\langle keys \rangle$.

```
905 \cs_new_protected:Nn \__enumext_before_keys_exec_vii:
906 {
907 \tl_use:N \l__enumext_before_no_starred_key_vii_tl
```

```
908  }
909  \cs_new_protected:Nn \__enumext_before_keys_exec_viii:
910   {
911     \tl_use:N \l__enumext_before_no_starred_key_viii_tl
912  }
```

The function __enumext_after_stop_list: executes the $\{\langle code \rangle\}$ set by the after key "after" the keyans environment has finished.

```
913 \cs_new_protected:Nn \__enumext_after_stop_list_vii:
914 {
915     \tl_use:N \l__enumext_after_stop_list_vii_tl
916    }
917 \cs_new_protected:Nn \__enumext_after_stop_list_viii:
918    {
919     \tl_use:N \l__enumext_after_stop_list_viii_tl
920    }
```

The function __enumext_after_args_exec_v: executes the $\{\langle code \rangle\}$ set by the first key after the end of $\{\langle arg\ two \rangle\}$ of the list defining the keyans environment, just before the first occurrence of \item.

```
921 \cs_new_protected:Nn \__enumext_after_args_exec_vii:
922 {
923    \tl_use:N \l__enumext_after_list_args_vii_tl
924  }
925 \cs_new_protected:Nn \__enumext_after_args_exec_viii:
926  {
927    \tl_use:N \l__enumext_after_list_args_viii_tl
928 }
```

 $(\textit{End of definition for } \verb|_-enumext_before_args_exec_vii: and others.)$

11.17 Setting keys for multicols and minipage

mini-env mini-sep columns-sep The default value of the columns-sep key is handled by the state of the boolean variable $\lower lambda$ columns_sep_X_bool which is handled in the internal definition of the enumext and keyans environments.

columns Define and set mini-env, mini-sep, columns-sep and columns keys for enumext and keyans environments.

```
929 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
930
931
      \keys_define:nn { enumext / #1 }
        {
932
                      .dim_set:c = { l__enumext_minipage_right_#2_dim },
          mini-env
                     .value_required:n = true,
          mini-env
                     .dim_set:c = { l__enumext_minipage_hsep_#2_dim },
          mini-sep
                     .initial:n = 0.3333em,
         mini-sep
         mini-sep
                     .value_required:n = true,
937
          columns-sep .dim_set:c = { l__enumext_columns_sep_#2_dim },
938
          columns-sep .value_required:n = true,
                    .int_set:c = { l__enumext_columns_#2_int },
          columns
          columns
                     .initial:n = 1,
          columns
                     .value_required:n = true,
945 \clist_map_inline:Nn \c__enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

For enumext* and keyans* environments the situation is a bit different, the default value for columns key are 2 and the command \miniright is not available, so we will add the keys miniright and miniright* to implement support for minipage.

```
946 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
947
       \keys_define:nn { enumext / #1 }
         {
                      .initial:n = 2,
           columns
           miniright .tl_gset:c = { g__enumext_miniright_code_#2_tl },
051
           miniright .value_required:n = true,
952
           miniright* .code:n
953
                                       \bool_gset_true:c { g__enumext_minipage_center_#2_bool }
954
                                       \keys_set:nn { enumext / #1 } { miniright = {##1} }
955
           miniright* .value_required:n = true,
957
_{99} \clist_map_inline:nn { {enumext*}{vii}, {keyans*}{viii} } { \__enumext_tmp:nn #1 }
©2024 by Pablo González L
```

42 / 125

(End of definition for mini-env and others.)

11.18 Adjustment of vertical spaces for multicols

When nesting a "list environment" inside the multicols environment, the values of the "vertical spaces" are lost, basically the multicols environment takes control over them. Graphically it can be seen like in the figure 7.



Figure 7: Representation of the vertical space in multicols for a nested level.

To keep the desired spaces *above* and *below* in the "*list environment*" (\topsep + [\partopsep]) it is necessary to "*adjust*" the spaces added by the multicols environment. The most appropriate option in this case is to use a "*context sensitive*" vertical space with \addvspace.

I should make it clear that the implementation here is a "bit questionable". At first glance doing \multicolsep=\topsep seemed right, but the results were not always as expected. An almost imperceptible detail is that in some cases the \itemsep values of are "stretched", possibly due to the use of \raggedcolumns and this affects the lower space when closing the environment, which is "smaller" than expected. My attempts to find the correct values using \showoutput and \showboxdepth absolutely failed.

11.18.1 Adjustment of vertical spaces for multicols in enumext

__enumext_multi_set_vskip:

The function __enumext_multi_set_vskip: will take care of determining the "adjusted spaces" that we will apply "above" and "below" the multicols environment in enumext.

We will set the default values taking into account that TEX is in $\langle horizontal \ mode \rangle$, then we will make the settings for the $\langle vertical \ mode \rangle$ in which $\langle partopsep \ comes$ into play.

Set the values of \l_enumext_multicols_above_X_skip and \l_enumext_multicols_below_-X_skip equal to the value of \topsep in the *current level*.

```
% \cs_new_protected:Nn \__enumext_multi_set_vskip:
% \skip_set:cn { l__enumext_multicols_above_ \__enumext_level: _skip }
% \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
% \skip_set:cn { l__enumext_multicols_below_ \__enumext_level: _skip }
% \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
% \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
% \__enumext_add_pre_parsep:
% }
```

(End of definition for __enumext_multi_set_vskip:.)

©2024 by Pablo González L

__enumext_add_pre_parsep:

The function $_$ _enumext_add_pre_parsep: "adjusted" the value of $_$ _enumext_multicols_above_X_skip detecting the value of $_$ parsep from the previous level. This is necessary since $_$ parsep from the previous level affects the *vertical spaces*.

43 / 125

```
993
994 }
995 }
```

(End of definition for __enumext_add_pre_parsep:.)

__enumext_multi_addvspace:

The function __enumext_multi_addvspace: will apply the spaces set using \addvspace "above" the multicols environment in enumext, taking into account whether $T_{E}X$ is in $\langle horizontal\ mode \rangle$ or $\langle vertical\ mode \rangle$.

11.18.2 Adjustment of vertical spaces for multicols in keyans

__enumext_keyans_multi_set_vskip:
__enumext_keyans_multi_addvspace:

The function __enumext_keyans_multi_set_vskip: will take care of determining the "adjusted spaces" that we will apply "above" and "below" the multicols environment in keyans. The implementation of this function is the same as the one used in enumext.

```
\cs_new_protected:Nn \__enumext_keyans_multi_set_vskip:
1015
       \skip_set:Nn \l__enumext_multicols_above_v_skip
1017
           \l__enumext_topsep_v_skip
       \skip_set:Nn \l__enumext_multicols_below_v_skip
         {
            \l__enumext_topsep_v_skip
         }
  \cs_new_protected:Nn \__enumext_keyans_multi_addvspace:
       \__enumext_keyans_multi_set_vskip:
       \mode_if_vertical:T
1028
           \skip_add:Nn \l__enumext_multicols_above_v_skip
1031
               \skip_use:N \l__enumext_partopsep_v_skip
           \skip_add:Nn \l__enumext_multicols_below_v_skip
               \skip_use:N \l__enumext_partopsep_v_skip
             }
         }
1038
       \par\nopagebreak
       \addvspace{ \l__enumext_multicols_above_v_skip }
1040
1041
```

11.19 Adjustment of vertical spaces for minipage

When nesting a "list environment" within the minipage environment, the values of the "vertical spaces" are lost. Graphically it can be seen like in the figure 8.

Since we want to keep the "left" and "right" environments "aligned on top", preserving the \baselineskip and keep the desired "spaces" (\topsep + [\partopsep]) it is necessary to "adjust" the "vertical spaces" for minipage environments.

44/125



Figure 8: Representation of the minipage spacing adjustment for a nested level.

Here there are several complications that we must circumvent, the minipage environment eliminates the "top" spaces, the multicols environment can be nested in the minipage environment, the "top" and "bottom" spaces are affected when topsep=%pt and to this is added the \partopsep parameter that comes into action according to whether TeX is in \(\lambda \text{horizontal mode} \rangle \) or \(\text{vertical mode} \). Depending on these cases, small adjustments must be made using \vspace and \addvspace to obtain the "desired vertical spacing".

of Again I must make clear that the implementation here is a "bit questionable", but hunting the spaces (glue) produced by the minipage environment is quite complicated, even more if multicols it is nested. The setting of the values was more "trial and error" (aprox to \strutbox), using the help of the lua-visual-debug[12] package, again my attempts to find the correct values using \showoutput and \showboxdepth absolutely failed.

__enumext_mini_env*

Creates a __enumext_mini_env* environment (*custom version* of minipage) setting the \if@minipage switch to "false" to allow spaces at the "above" of the environment, plus we will add \vspace{\opt} to maintain alignment on "top". This environment will be used internally by the mini-env key, it is not documented in the user interface and is for internal use only.

 $(End\ of\ definition\ for \verb|_= enumext_mini_env*.)$

11.19.1 Adjustment of vertical spaces for minipage in enumext

__enumext_mini_set_vskip:

The function __enumext_mini_set_vskip: will take care of determining the "adjust" spaces that we will apply "above" and "below" the __enumext_mini_env* environment in enumext.

We will set the default values taking into account that TeX is in $\langle horizontal \ mode \rangle$, then we will make the settings for the $\langle vertical \ mode \rangle$ in which \rangle comes into play.

First determine if the multicols environment is active by comparing the value of the \l__enumext_-columns_X_int variable handled by the columns key, according to this comparison we set the adjusted values for \l_enumext_minipage_left_skip, \l_enumext_minipage_right_skip and \l_-enumext_minipage_after_skip.

If multicols environment is nested in __enumext_mini_env* environment, we will apply a correction factor to the *vertical spaces* taking into account the value of \topsep of the current level and the value of \parsep of the previous level, if these are zero we will use \strutbox as the basis for the calculations.

If only enumext environment is nested in __enumext_mini_env* environment, we will apply a correction factor to the *vertical spaces* taking into account the value of \topsep, if this is zero we will use \strutbox as the basis for the calculations.

```
\skip_if_eq:nnTF
             { \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip } } { \c_zero_skip }
               \skip_set:Nn \l__enumext_minipage_left_skip
                 {
                   0.5\box_dp:N \strutbox
                   - \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
               \skip_set:Nn \l__enumext_minipage_right_skip
                   \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
                 }
               \skip_set:Nn \l__enumext_minipage_after_skip
                   1.6\box_dp:N \strutbox
                 }
             }
               \skip set:Nn \l enumext minipage left skip
                 {
                   0.5875\box_dp:N \strutbox
                   - \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
                 }
               \skip_set:Nn \l__enumext_minipage_right_skip
                   + \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
                   + \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
               \skip_set:Nn \l__enumext_minipage_after_skip
1116
                {
                   0.325\box_dp:N \strutbox
1118
                    \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
             }
         }
```

 $(End\ of\ definition\ for\ \verb|__enumext_mini_set_vskip:.)$

__enumext_zero_parsep:

The function __enumext_zero_parsep: "adjusted" the value of \l__enumext_minipage_after_-skip detecting the value of \parsep from the previous level. This is necessary since \parsep from the previous level affects the vertical spaces and this is noticeable when using the nosep or noitemsep keys.

enumext mini addvspace:

The function __enumext_mini_addvspace: will apply the spaces set using \addvspace "above" the __enumext_mini_env* environment in enumext, taking into account whether TEX is in \langle horizontal mode \rangle or \langle vertical mode \rangle. For the latter we will make some adjustments since the \partopsep parameter comes into play and this affects the vertical spacing.

(End of definition for __enumext_mini_addvspace:.)

(End of definition for $\ensuremath{\backslash}$ enumext_zero_parsep:.)

11.19.2 Adjustment of vertical spaces for minipage in keyans

__enumext_keyans_mini_set_vskip:

The function __enumext_keyans_mini_set_vskip: will take care of determining the "adjusted" spaces that we will apply "above" and "below" the __enumext_mini_env* environment in keyans. The implementation of this function is the same as the one used in enumext.

```
\cs_new_protected:Nn \__enumext_keyans_mini_set_vskip:
1166
       \skip_zero_new:N \l__enumext_minipage_after_skip
1167
       \skip_zero_new:N \l__enumext_minipage_left_skip
1168
       \skip_zero_new:N \l__enumext_minipage_right_skip
1169
       \int_compare:nNnTF { \l__enumext_columns_v_int } > { 1 }
           \skip_if_eq:nnTF { \l__enumext_topsep_v_skip } { \c_zero_skip }
             {
               \skip_set:Nn \l__enumext_minipage_left_skip { -0.25\box_dp:N \strutbox }
               \skip_set:Nn \l__enumext_minipage_right_skip { 0.705\box_dp:N \strutbox }
               \skip_set:Nn \l__enumext_minipage_after_skip { \box_dp:N \strutbox }
               \skip_if_eq:nnF { \l__enumext_parsep_i_skip } { \c_zero_skip }
                 {
1178
                   \skip_add:Nn \l__enumext_minipage_after_skip { 2.15\box_dp:N \strutbox }
1180
             }
1181
               \skip_set:Nn \l__enumext_minipage_left_skip
                   \skip_use:N \l__enumext_topsep_v_skip
               \skip_set:Nn \l__enumext_minipage_right_skip
1187
```

```
0.705\box_dp:N \strutbox
                 }
               \skip_set:Nn \l__enumext_minipage_after_skip
                   1.85\box_dp:N \strutbox + \l__enumext_topsep_v_skip
1194
             }
         }
           \skip_if_eq:nnTF { \l__enumext_topsep_v_skip } { \c_zero_skip }
               \skip_set:Nn \l__enumext_minipage_left_skip
                 {
                   0.5\box_dp:N \strutbox
                   + \l enumext partopsep v skip
1203
               \skip_set:Nn \l__enumext_minipage_right_skip
                 {
1206
                    \l__enumext_partopsep_v_skip
               \skip_set:Nn \l__enumext_minipage_after_skip { 1.6\box_dp:N \strutbox }
             }
               \skip_set:Nn \l__enumext_minipage_left_skip
                 {
                   0.5875\box_dp:N \strutbox - \l__enumext_partopsep_v_skip
1214
               \skip_set:Nn \l__enumext_minipage_right_skip
                 {
                   \l__enumext_topsep_v_skip + \l__enumext_partopsep_v_skip
                 }
               \skip_set:Nn \l__enumext_minipage_after_skip
                 {
                   0.325\box_dp:N \strutbox + \l__enumext_topsep_v_skip
                 }
1223
             }
         }
1226
```

__enumext_keyans_mini_addvspace:

The function __enumext_keyans_mini_addvspace: will apply the spaces set using \addvspace "above" the __enumext_mini_env* environment in keyans, taking into account whether TeX is in $\langle horizontal\ mode \rangle$ or $\langle vertical\ mode \rangle$. For the latter we will make some adjustments since the \partopsep parameter comes into play and this affects the $vertical\ spacing$. The implementation of this function is the same as the one used in enumext.

```
\label{local_local_local_local_local} $$ \cs_new\_protected:Nn \ \_enumext_keyans\_mini\_addvspace: $$
1228
                                                     \__enumext_keyans_mini_set_vskip:
                                                   \mode_if_vertical:T
 1230
 1231
                                                                                \skip_add:Nn \l__enumext_minipage_left_skip
 1232
                                                                                                              \label{local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_local_loc
                                                                                \skip_add:Nn \l__enumext_minipage_after_skip
                                                                                                                \l__enumext_partopsep_v_skip
 1238
 1240
                                                   \par\nopagebreak
 1241
                                                   \addvspace { \l__enumext_minipage_left_skip }
 1242
                                    }
```

(End of definition for __enumext_keyans_mini_addvspace:.)

(End of definition for __enumext_keyans_mini_set_vskip:.)

11.19.3 Adjustment of vertical spaces for minipage in enumext* and keyans*

__enumext_mini_set_vskip_vii:
__enumext_mini_set_vskip_viii:

The functions __enumext_mini_set_vskip_vii: and __enumext_mini_set_vskip_viii: will take care of determining the "adjusted" spaces that we will apply "above" and "below" the __enumext_mini_env* environment in enumext* and keyans*.

```
\cs_new_protected:Nn \__enumext_mini_set_vskip_vii:
1245
       \skip_zero_new:N \l__enumext_minipage_left_skip
1246
       \skip_gzero_new:N \g__enumext_minipage_right_skip
1247
       \skip_gzero_new:N \g__enumext_minipage_after_skip
1248
       \skip_if_eq:nnTF { \l__enumext_topsep_vii_skip } { \c_zero_skip }
1249
            \skip_set:Nn \l__enumext_minipage_left_skip { 0.5\box_dp:N \strutbox }
            \skip_gset:Nn \g__enumext_minipage_right_skip { 0.325\box_dp:N \strutbox }
         }
         {
            \skip_set:Nn \l__enumext_minipage_left_skip { 0.5875\box_dp:N \strutbox }
            \skip_gset:Nn \g__enumext_minipage_right_skip
             {
                \l__enumext_topsep_vii_skip
1258
            \skip_gset:Nn \g__enumext_minipage_after_skip
1260
             {
1261
                0.325\box_dp:N \strutbox + \l__enumext_topsep_vii_skip
1262
         }
   \cs_new_protected:Nn \__enumext_mini_set_vskip_viii:
1267
       \skip_zero_new:N \l__enumext_minipage_after_skip
1268
       \skip_zero_new:N \l__enumext_minipage_left_skip
1269
       \skip_zero_new:N \l__enumext_minipage_right_skip
       \skip_if_eq:nnTF { \l__enumext_topsep_viii_skip } { \c_zero_skip }
            \skip_set:Nn \l__enumext_minipage_left_skip
1274
                0.5\box_dp:N \strutbox
             1
            \skip_set:Nn \l__enumext_minipage_right_skip
             {
                \verb|\l_enumext_partopsep_viii_skip|
1280
            \skip_set:Nn \l__enumext_minipage_after_skip
1281
             {
1282
                1.6\box_dp:N \strutbox
1283
         }
         {
            \skip_set:Nn \l__enumext_minipage_left_skip
                0.5875\box_dp:N \strutbox
            \skip_set:Nn \l__enumext_minipage_right_skip
                \l__enumext_topsep_viii_skip
1293
            \skip_set:Nn \l__enumext_minipage_after_skip
                0.325\box_dp:N \strutbox + \l__enumext_topsep_viii_skip
1298
          }
     }
1300
(End of definition for \__enumext_mini_set_vskip_vii: and \__enumext_mini_set_vskip_viii:.)
```

__enumext_mini_addvspace_vii:
__enumext_mini_addvspace_viii:

The functions __enumext_mini_addvspace_vii: and __enumext_mini_addvspace_viii: will apply the vertical space "only above" the __enumext_mini_env* environment on the left side when the miniright key is active in the enumext* and keyans* environments.

Here we will NOT take into account whether T_EX is in $\langle horizontal\ mode \rangle$ or $\langle vertical\ mode \rangle$, since $\langle partopsep$ is equal to opt in both environments.

```
1301 \cs_new_protected:Nn \__enumext_mini_addvspace_vii:
1302 {
1303 \__enumext_mini_set_vskip_vii:
1304 \par\nopagebreak
1305 \addvspace { \l__enumext_minipage_left_skip }
1306 }

©2024 by Pablo González L
```

```
\cs_new_protected:Nn \__enumext_mini_addvspace_viii:
1308 {
1309 \__enumext_mini_set_vskip_viii:
1310 \par\nopagebreak
1311 \addvspace { \l__enumext_minipage_left_skip }
1312 }
```

 $(\textit{End of definition for $_=$enumext_mini_addvspace_vii: and $_=$enumext_mini_addvspace_viii:.)$}$

11.19.4 The command \miniright

The command \miniright will close the __enumext_mini_env* environment on the "left side", open the __enumext_mini_env* environment on the "right side" adding the adjusted vertical space. By default we will add \centering when starting the "right side" environment. The starred version '*' inhibits the use of \centering command i.e. the usual ETEX justification is maintained in the __enumext_mini_env* on the "right side".

\miniright

First we will perform some checks to prevent the command from being executed outside the enumext environment or from being executed inside the keyanspic environment, then we call the internal functions for the enumext and keyans environments.

(End of definition for \miniright. This function is documented on page 9.)

_enumext_mini_right_cmd:n

The function __enumext_mini_right_cmd:n takes as argument the *starred version* '*' of the \miniright command in the enumext environment. We check if the mini-env key is active via the variable \l__-enumext_minipage_right_X_dim, if so we close the multicols environment with the __enumext_mini_env* environment on the "left side", then we open the __enumext_mini_env* environment on the "right side", apply our adjusted "vertical spaces", followed by adding the \centering command when the starred argument '*' is not present and set zero \g__enumext_minipage_stat_int, otherwise we return an error.

```
\cs_new_protected:Npn \__enumext_mini_right_cmd:n #1
       \dim compare:nNnTF
1331
         { \dim_use:c { l__enumext_minipage_right_ \__enumext_level: _dim } } > { \c_zero_dim }
           \ enumext multicols stop:
           \end{__enumext_mini_env*}
           \hfill
           \begin{__enumext_mini_env*}
             { \dim_use:c { l__enumext_minipage_right_ \__enumext_level: _dim } }
1338
             \par\addvspace { \l__enumext_minipage_right_skip }
             \bool_if:nF {#1}
               {
                 \centering
             \int_gzero:N \g__enumext_minipage_stat_int
         }
1345
         { \msg_error:nnn { enumext } { wrong-miniright-use } }
1346
1347
```

 $(End\ of\ definition\ for\ \verb|_-enumext_mini_right_cmd:n.|)$

__enumext_keyans_mini_right_cmd:n

The function __enumext_keyans_mini_right_cmd:n takes as argument the *starred version* '*' of the \miniright command in the keyans environment. The implementation of this function is the same as that of the __enumext_mini_right_cmd:n function of the enumext environment.

```
\cs_new_protected:Npn \__enumext_keyans_mini_right_cmd:n #1
1349
       \dim_compare:nNnTF { \l__enumext_minipage_right_v_dim } > { \c_zero_dim }
1350
1351
           \__enumext_keyans_multicols_stop:
           \end{__enumext_mini_env*}
           \hfill
           \begin{__enumext_mini_env*}{ \l__enumext_minipage_right_v_dim }
             \par\addvspace { \l__enumext_minipage_right_skip }
             \bool_if:nF {#1}
               {
1358
                 \centering
1360
             \int_gzero:N \g__enumext_minipage_stat_int
1361
         { \msg_error:nnn { enumext } { wrong-miniright-use } }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|__enumext_keyans_mini_right_cmd:n.)$

11.20 Setting above and below keys

While having controlled the *vertical spaces* within the enumext and keyans environments when using the columns or mini-env keys, sometimes the "vertical spaces above" or "vertical spaces below" the environments are not as expected and it is necessary to be able to apply a "fine correction" to these. As I have not been able to correct these *glitches*, the best option is to leave a couple of $\langle keys \rangle$ dedicated to this purpose, in this case it is best to use \vspace or \vspace* when convenient.

```
above
        Define above, above*, below and below* keys for enumext and keyans environments.
above*
        1365 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
 below
helow*
               \keys_define:nn { enumext / #1 }
        1367
                    above .skip_set:c = { l__enumext_vspace_above_#2_skip },
                    above .value_required:n = true,
        1370
                    above* .code:n
                                       = \bool_set_true:c { l__enumext_vspace_a_star_#2_bool }
                                         \keys_set:nn { enumext / #1 } { above = {##1} },
                    above* .value_required:n = true,
                          .skip_set:c = { l__enumext_vspace_below_#2_skip },
                   below
                          .value_required:n = true,
                   below* .code:n
                                       = \bool_set_true:c { l__enumext_vspace_b_star_#2_bool }
                                         \keys_set:nn { enumext / #1 } { below = {##1} },
                   below* .value_required:n = true,
                 }
        1379
        1380
        1381 \clist_map_inline:Nn \c__enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

(End of definition for above and others.)

11.20.1 Functions for above and below keys in enumext

__enumext_vspace_above:

The function __enumext_vspace_above: apply the *vertical space above* the enumext environment set by the above* and above keys.

```
\cs_new_protected:Nn \__enumext_vspace_above:
1383
       \skip_if_eq:nnF
1384
         { \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } } { \c_zero_skip }
1385
1386
           \bool_if:cTF { l__enumext_vspace_a_star_ \__enumext_level: _bool }
1387
1388
                \vspace*{ \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } }
1389
             }
             {
                \vspace { \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } }
         }
1394
1395
```

 $(End\ of\ definition\ for\ \verb|__enumext_vspace_above:.)$

__enumext_vspace_below:

The function __enumext_vspace_below: apply the *vertical space below* the enumext environment set by the below* and below keys.

(End of definition for $__$ enumext $_$ vspace $_$ below:.)

11.20.2 Functions for above and below keys in keyans

__enumext_vspace_above_v:

The function __enumext_vspace_above_v: apply the *vertical space above* the keyans environment set by the above and above* keys.

 $(\mathit{End}\ of\ definition\ for\ \verb|__enumext_vspace_above_v:.)$

__enumext_vspace_below_v:

The function __enumext_vspace_below_v: apply the *vertical space below* the keyans environment set by the below* and below keys.

(End of definition for __enumext_vspace_below_v:.)

11.20.3 Functions for above and below keys in enumext* keyans*

__enumext_vspace_above_vii:
\ enumext vspace above viii:

The functions __enumext_vspace_above_vii: and __enumext_vspace_above_viii: apply the vertical space above the enumext* and keyans* environments set by the above and above* keys.

```
\cs_new_protected:Nn \__enumext_vspace_above_vii:
    {
       \skip_if_eq:nnF { \l__enumext_vspace_above_vii_skip } { \c_zero_skip }
1434
1435
           \bool_if:NTF \l__enumext_vspace_a_star_vii_bool
1437
                \vspace*{ \l__enumext_vspace_above_vii_skip }
1438
1439
             { \vspace { \l__enumext_vspace_above_vii_skip } }
1441
   \cs_new_protected:Nn \__enumext_vspace_above_viii:
1445
       \skip_if_eq:nnF { \l__enumext_vspace_above_viii_skip } { \c_zero_skip }
1446
         {
```

 The functions __enumext_vspace_below_vii: and __enumext_vspace_below_viii: apply the vertical space below the enumext* and keyans* environments set by the below* and below keys.

```
\cs_new_protected:Nn \__enumext_vspace_below_vii:
       \skip_if_eq:nnF { \l__enumext_vspace_below_vii_skip } { \c_zero_skip }
           \bool_if:NTF \l__enumext_vspace_b_star_vii_bool
             {
               \vspace*{ \l__enumext_vspace_below_vii_skip }
1461
             { \vspace { \l__enumext_vspace_below_vii_skip } }
1462
1463
1464
   \cs_new_protected:Nn \__enumext_vspace_below_viii:
       \skip_if_eq:nnF { \l__enumext_vspace_below_viii_skip } { \c_zero_skip }
1467
           \bool_if:NTF \l__enumext_vspace_b_star_viii_bool
             {
               \vspace*{ \l__enumext_vspace_below_viii_skip }
1471
             { \vspace { \l__enumext_vspace_below_viii_skip } }
1473
         }
1474
     }
1475
```

 $(\textit{End of definition for } \verb|_=enumext_vspace_below_vii: and \verb|_=enumext_vspace_below_viii:|)$

11.21 Setting series, resume and resume* keys

The series key is responsible for the whole process of the resume and resume* keys. The idea behind this is to be able to absorb the $\langle keys \rangle$ passed to the optional argument of the "first level" of the environments enumext and enumext*, but, discarding some specific $\langle keys \rangle$. This implementation is adapted directly from the code provided by Jonathan P. Spratte (@Skillmon) in chat-TeX-SX

We define the keys series, resume and resume* only for the "first level" of enumext and enumext*.

```
resume
         1476 \cs_set_protected:Npn \__enumext_tmp:n #1
resume*
              {
         1477
                 \keys_define:nn { enumext / #1 }
         1478
                   {
         1479
                     series .str_set:N = \l__enumext_series_str,
                     series .value_required:n = true,
                     resume .code:n = \__enumext_resume_series:n {##1},
                     resume* .code:n = \__enumext_resume_starred:,
                     resume* .value_forbidden:n = true,
                   }
         1485
         1486
         1487 \clist_map_inline:nn { level-1, enumext* } { \__enumext_tmp:n {#1} }
```

(End of definition for series, resume, and resume * .)

11.21.1 Internal functions for series key

__enumext_filter_series:n __enumext_filter_series_key:n __enumext_filter_series_pair:nn

series

The function $_$ enumext_filter_series:n will be in charge of filtering the $\langle keys \rangle$ we want to store where $\{\#1\}$ represents the optional value passed to the environment.

The function __enumext_filter_series_key:n will be responsible for filtering the $\langle keys \rangle$ that are passed "without value" by excluding the resume and resume* keys.

The function __enumext_filter_series_pair:nn will be responsible for filtering the $\langle keys \rangle$ that are passed "with value" by excluding the series, resume, start, save-ans and save-key keys.

 $(\textit{End of definition for } \climate{line} enumext_filter_series:n, \climate{line} enumext_filter_series_key:n, and \climate{line} enumext_filter_series_pair:nn.)$

__enumext_parse_series:n
\ enumext resume last:n

The function __enumext_parse_series:n will be responsible for storing the filtered $\langle keys \rangle$ in the global variable \g__enumext_series_ $\langle series\ name \rangle$ _tl along with the creation of the integer variable \g__enumext_series_ $\langle series\ name \rangle$ _int when the key is passed as an argument; otherwise, it will check the state of the boolean variable \l_enumext_resume_active_bool set by the keys resume and resume* and will call the function _enumext_resume_last:n.

The value of boolean variable \l__enumext_resume_active_bool is set to true by the function __enumext_resume_counter:n which is used by the keys resume and resume*, in this case we must Make sure it is set to false so that it does not overwrite the default filtered \(\lambda eys \rangle \). This function is passed to the function __enumext_parse_keys:n in the enumext environment definition (\(\subseteq 11.31 \)) and to the function __enumext_parse_keys_vii:n in the enumext* environment definition (\(\subseteq 11.34 \)).

```
\cs_new_protected:Npn \__enumext_parse_series:n #1
    {
       \str_if_empty:NTF \l__enumext_series_str
           \bool_if:NF \l__enumext_resume_active_bool
                __enumext_resume_last:n {#1}
         }
1526
           \tl_gclear_new:c { g__enumext_series_ \l__enumext_series_str _tl }
1528
           \tl_gset:ce { g__enumext_series_ \l__enumext_series_str _tl }
             { \__enumext_filter_series:n {#1} }
           \int_if_exist:cF { g__enumext_series_ \l__enumext_series_str _int }
               \int_new:c { g__enumext_series_ \l__enumext_series_str _int }
             }
1534
         }
1536
```

The function __enumext_resume_last:n will be in charge of saving the filtering $\langle keys \rangle$ when the series key is *not used* and will save them in the variable \g__enumext_standar_series_tl for the enumext environment and in the variable \g__enumext_starred_series_tl for the enumext* environment. Here we must use \bool_lazy_all:nT to make sure that the default values are not overwritten when the environment is nested and the series key is not being used.

(End of definition for __enumext_parse_series:n and __enumext_resume_last:n.)

11.21.2 Internal function to save counter value

__enumext_resume_save_counter:

The __enumext_resume_save_counter: function will save the last counter value to \g__enumext_-series_ $\langle series\ name \rangle$ _int if the series= $\{\langle series\ name \rangle\}$ key has been passed, to \g__enumext_-resume_int if it has passed the key resume without value and the key series is not active, in \g__-enumext_series_ $\langle series\ name \rangle$ _int if the key resume= $\{\langle series\ name \rangle\}$ has been passed and in \g_-enumext_series_ $\langle store\ name \rangle$ _int if the key has been passed save-ans= $\{\langle store\ name \rangle\}$.

The variables \l__enumext_series_str and \l__enumext__resume_name_tl contain the same {\series name\} but are executed at different moments, the integer variable with \l__enumext_series_str sets the value when execute series={\series name\} and the integer variable with \l__enumext__resume_name_tl sets the subsequent values when use resume={\series name\}. This function is passed to the enumext environment definition (\§11.31) and the enumext* environment definition (\§11.34).

```
\cs_new_protected:Nn \__enumext_resume_save_counter:
1551
     {
       \bool_if:NT \g__enumext_standar_bool
           \tl_if_empty:NF \l__enumext_series_str
1554
             {
                \int_gset_eq:cN
1556
                  { g__enumext_series_ \l__enumext_series_str _int } \value{enumXi}
           \tl_if_empty:NTF \l__enumext_resume_name_tl
             {
                \str_if_empty:NT \l__enumext_series_str
1561
1562
                    \int_gset_eq:NN \g__enumext_resume_int \value{enumXi}
1563
1564
             }
1565
1566
                \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
1567
                  {
                    \int_gset_eq:cN
                      { g__enumext_series_ \l__enumext_resume_name_tl _int } \value{enumXi}
             }
           \int_if_exist:cT { g__enumext_resume_ \l__enumext_store_name_tl _int }
             {
1574
                \int_gset_eq:cN
                  { g__enumext_resume_ \l__enumext_store_name_tl _int } \value{enumXi}
1576
         }
       \bool_if:NT \g__enumext_starred_bool
           \tl_if_empty:NF \l__enumext_series_str
1582
                \int_gset_eq:cN
1583
                  { g__enumext_series_ \l__enumext_series_str _int } \value{enumXvii}
1584
1585
           \tl_if_empty:NTF \l__enumext_resume_name_tl
1586
             {
1587
                \str_if_empty:NT \l__enumext_series_str
                    \int_gset_eq:NN \g__enumext_resume_vii_int \value{enumXvii}
                  }
             }
1593
                \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
1594
                  {
1595
                    \int_gset_eq:cN
1596
                      { g__enumext_series_ \l__enumext_resume_name_tl _int } \value{enumXvii}
1597
1598
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_resume_save_counter:.)$

11.21.3 Internal functions for resume key

__enumext_resume_series:n

The function __enumext_resume_series:n will handle the argument passed to the resume key in enumext and enumext* environments. If the key is passed without value the function __enumext_resume_counter: is executed which will set the counter according to the numbering of the last enumext or enumext* environments in which series={\langle series name \rangle} key is not present, if the save-ans key is active it will set the counter according to the value of the integer variable created by that key, otherwise it will verify that the \g__enumext_series_\langle series name \rangle_tl variable set by the series key exists, if so it will pass these keys to the first level of the environment, otherwise it will return an error.

```
1607 \cs_new_protected:Npn \__enumext_resume_series:n #1
1608
       \tl_if_empty:nTF {#1}
1609
         {
1610
             _enumext_resume_counter:n { }
1611
         }
1612
         {
1613
           \tl_if_exist:cTF { g__enumext_series_ \tl_to_str:n {#1} _tl }
                \__enumext_resume_counter:n {#1}
                \bool_if:NT \g__enumext_standar_bool
                    \keys_set:nv { enumext / level-1 }
                      { g__enumext_series_ \tl_to_str:n {#1} _tl }
                  }
                \bool_if:NT \g__enumext_starred_bool
                    \keys_set:nv { enumext / enumext* }
                      { g__enumext_series_ \tl_to_str:n {#1} _tl }
                  }
             }
                \bool_if:NT \g__enumext_standar_bool
                  {
                    \msg_error:nnn { enumext } { unknown-series } {#1}
1631
1632
                \bool_if:NT \g__enumext_starred_bool
1633
                  {
1634
                    \msg_error:nnn { enumext } { unknown-series } {#1}
         }
```

(End of definition for __enumext_resume_series:n.)

__enumext_resume_counter:n
__enumext_resume_counter_series:
_enumext_resume_counter_save_ans:

The function __enumext_resume_counter:n will set the variable \l__enumext_resume_active_bool to true and pass the value of the key resume to the variable \l__enumext_series_name_tl which will contain the $\{\langle series\ name \rangle\}$. If the variable \l__enumext_series_name_tl is empty, that is, we are passing the key resume without value, we will execute the function __enumext_resume_counter: otherwise, when we pass resume= $\{\langle series\ name \rangle\}$ we will execute the function __enumext_resume_counter_series:, finally we will execute the function __enumext_resume_counter_save_ans: which is associated with the key save-ans.

```
1640 \cs_new_protected:Npn \__enumext_resume_counter:n #1
1641 {
1642    \bool_set_true:N \l__enumext_resume_active_bool
1643    \tl_set:Nn \l__enumext_resume_name_tl {#1}
1644    \tl_if_empty:NTF \l__enumext_resume_name_tl
1645    {
1646    \__enumext_resume_counter:
1647    }
```

The __enumext_resume_counter: function is executed when the resume key is used without value, only the counters for the "first level" of the environments will be set.

```
\cs_new_protected:Nn \__enumext_resume_counter:
1654
       \verb|\bool_if:NT \g_enumext_standar_bool|\\
1655
1656
            \int_gincr:N \g__enumext_resume_int
1657
            \int_set_eq:NN \l__enumext_start_i_int \g__enumext_resume_int
1658
1659
       \bool_if:NT \g__enumext_starred_bool
1660
            \int_gincr:N \g__enumext_resume_vii_int
            \int_set_eq:NN \l__enumext_start_vii_int \g__enumext_resume_vii_int
1663
         }
1665
```

The function __enumext_resume_counter_series: will be executed when the resume= $\{\langle series name \rangle\}$ key is active, setting the counters for the "first level" of the environments according to the value of the integer variables created by the series key.

```
\cs_new_protected:Nn \__enumext_resume_counter_series:
1667
       \bool_if:NT \g__enumext_standar_bool
1668
1669
           \int_set:Nn \l__enumext_start_i_int
                \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
       \bool_if:NT \g__enumext_starred_bool
           \int_set:Nn \l__enumext_start_vii_int
1677
1678
             {
                \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
1679
1680
         }
1681
1682
```

The function __enumext_resume_counter_save_ans: will be executed when the save-ans key is active along with the resume key, setting the counters for the "first level" of the environments according to the value of the integer variables created by the save-ans key.

```
\cs_new_protected:Nn \__enumext_resume_counter_save_ans:
       \bool_lazy_and:nnT
         { \bool_if_p:N \l__enumext_standar_first_bool }
         { \bool_if_p:N \l__enumext_store_active_bool }
1687
1688
           \int_set:Nn \l__enumext_start_i_int
1689
               \int_use:c { g__enumext_resume_ \l__enumext_store_name_tl _int } + 1
         }
       \bool_lazy_and:nnT
         { \bool_if_p:N \l__enumext_starred_first_bool }
         { \bool_if_p:N \l__enumext_store_active_bool }
           \int_set:Nn \l__enumext_start_vii_int
1698
               \int_use:c { g__enumext_resume_ \l__enumext_store_name_tl _int } + 1
1701
         }
1702
     }
1703
```

 $(\textit{End of definition for } \verb|_-enumext_resume_counter:n and others.)$

11.21.4 Internal function for resume* key

__enumext_resume_starred:

The function $_$ _enumext_resume_starred: will handle the resume* key in the enumext and enumext* environments. This function will execute the filtered $\langle keys \rangle$ in the last one and will continue with the numbering according to the last execution of the environment enumext or enumext* in which the keys resume= $\{\langle series\ name \rangle\}$ or series= $\{\langle series\ name \rangle\}$ were not active.

(End of definition for __enumext_resume_starred:.)

11.22 Setting save-ans, check-ans and no-store keys

The key save-ans is directly associated with the keys check-ans, no-store, resume and resume*, this will activate the entire "storage system" in the enumext package.

11.22.1 Setting save-ans key

save-ans We define the keys save-ans only for the "first level" of enumext and enumext*.

(End of definition for save-ans.)

11.22.2 Internal functions for save-ans key

__enumext_start_save_ans_msg:
\ enumext stop save ans msg:

The functions __enumext_start_save_ans_msg: and __enumext_stop_save_ans_msg: will display in the terminal and .log file the environment in which the save-ans key was executed along with the line at the beginning and end of it. The function __enumext_start_save_ans_msg: will be passed to __enumext_storing_set:n and the function __enumext_stop_save_ans_msg: will be passed to the function __enumext_execute_after_env:.

__enumext_storing_set:n
__enumext_storing_exec:

The function __enumext_storing_set:n first pass the value of the save-ans key to the variable \l__enumext_store_name_tl which will contain the "store name" of the $\langle sequence \rangle$ and $\langle prop \ list \rangle$ we will use. If \l__enumext_store_name_tl is empty we return an error message, otherwise will return the appropriate message __enumext_start_save_ans_msg: and proceed to execute the function __enumext_storing_exec: for enumext and enumext* environments.

```
\cs_new_protected:Npn \__enumext_storing_set:n #1
1743
       \tl set:Ne \l enumext store name tl {#1}
1744
       \tl_if_empty:NTF \l__enumext_store_name_tl
1745
1746
           \bool_lazy_or:nnT
1747
             { \l_enumext_standar_first_bool } { \l_enumext_starred_first_bool }
1748
1749
                \msg_error:nnV { enumext } { save-ans-empty } \g_enumext_envir_name_tl
         }
         {
           \bool_lazy_or:nnT
             { \l__enumext_standar_first_bool } { \l__enumext_starred_first_bool }
                \ enumext start save ans msg:
                \__enumext_storing_exec:
1758
         }
1760
1761
```

The function __enumext_storing_exec: will set to true the variable \l__enumext_store_active_bool which activates the use of the \anskey command and the keyans, keyans* and keyanspic environments and will set to true the variable \l__enumext_check_answers_bool used for checking answers by the check-ans and no-store keys. The $\langle prop\ list\rangle$ \g__enumext_series_ $\langle store\ name\rangle$ _prop and the $\langle sequence\rangle$ \g_enumext_series_ $\langle store\ name\rangle$ _seq will be created globally to "store content" in case they do not exist together with the integer variable \g_enumext_series_ $\langle store\ name\rangle$ _int used by the keys resume and resume*.

```
1762 \cs_new_protected:Nn \__enumext_storing_exec:
1763
    {
       \bool_set_true:N \l__enumext_store_active_bool
1764
       \bool_set_true:N \l__enumext_check_answers_bool
1765
       \tl_gset:NV \g__enumext_store_name_tl \l__enumext_store_name_tl
1766
       \prop_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _prop }
1767
1768
           \msg_log:nnV { enumext } { store-prop } \l__enumext_store_name_tl
1769
           \prop_new:c { g__enumext_ \l__enumext_store_name_tl _prop }
         }
       \seq_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _seq }
1773
         {
           \msg_log:nnV { enumext } { store-seq } \l__enumext_store_name_tl
           \seq_new:c { g__enumext_ \l__enumext_store_name_tl _seq }
         }
       \int_if_exist:cF { g__enumext_resume_ \l__enumext_store_name_tl _int }
1778
           \msg_log:nnV { enumext } { store-int } \l__enumext_store_name_tl
1780
           \int_new:c { g__enumext_resume_ \l__enumext_store_name_tl _int }
1781
         }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_=enumext_storing_set:n\ and\ \verb|_=enumext_storing_exec:|)$

11.22.3 The check answer mechanism

The mechanism for checking that all questions are answered follows this logic:

If the line begins with \item or \item* and does NOT open a nested environment, each \item or \item* must contain a single execution of the \anskey command, i.e. the counter of the executions of the \anskey command must be equal to the counter associated with the sum of executions of \item and \item*.

If the line begins with \item or \item* and opens a nested environment each \item or \item* in the nested environment must have a single execution of the \anskey command and the counter associated to the sum of \item and \item* executions must decrementing by "one" to maintain equality.

In order for the mechanism for the check-answer to work (not counting keyans, keyans* and keyanspic) we need:

- 1. We must keep track of the total number of \item and \item* (enumerated) that appear within the environment including the nested levels.
- 2. We must keep track of the total number of \item and \item* (enumerated) that appear per level of nesting.

3. Keeping track of the number of times the environment nests.

The integer variable associated to the sum of each $\idesigned item$ and $\idesigned integer variable <math>\g_{enumext_item}$ in the environment $\g_{enumext_item}$ anskey_int associated to the execution of the command \anskey . We analyze the cases:

- a) If the list only has one level the number of \item + \item* = \anskey
- b) If the list has *nested levels*, for each level of nesting we need to decrementing by one (for the \item or \item* that opens the nest) so that the account remains the same.

With keyans, keyans* and keyanspic it is enough to increase in one the integer of \anskey. The integers created must be global if they are not lost in the interior levels of nesting and to execute the test we will use a "hook" function after closing the first level of the environment.

11.22.4 Setting check-ans and no-store keys

check-ans

Now we define the keys check-ans and no-store for all levels of enumext and enumext* environments.

```
1783 \cs_set_protected:Npn \__enumext_tmp:n #1
       \keys_define:nn { enumext / #1 }
1785
         {
1786
           check-ans .bool_set:N = \l__enumext_check_ans_key_bool,
1787
           check-ans .initial:n = false,
1788
           check-ans .value_required:n = true,
1789
           no-store .code:n = {
                                    \bool_set_false:N \l__enumext_check_answers_bool
1791
                                   \bool_set_false:N \l__enumext_check_ans_key_bool
                                 },
           no-store .value_forbidden:n = true,
         }
1796
   \clist_map_inline:nn
1797
1798
     {
       level-1, level-2, level-3, level-4, enumext*
1799
1800
     { \__enumext_tmp:n {#1} }
```

 $(\mathit{End}\ of\ definition\ for\ check-ans\ \ and\ no\text{-store.})$

11.22.5 Set-up check answer mechanism

__enumext_check_ans_active:
__enumext_check_ans_level:

The function __enumext_check_ans_active: will first check the state of the variable \l__enumext_-store_name_tl, that is, the save-ans key is active, if so it will check the state of the variable \l__enumext_check_answers_bool handled by the key no-store and will execute the function __enumext_check_ans_level: only if "true", i.e. the key no-store is not active.

The function __enumext_check_ans_level: will decrement by "one" the value of the variable \g__-enumext_item_number_int which keeps track of the executions of \item and \item* for each level of nesting of the environment enumext, taking into account whether it is nested within enumext* or the opposite.

60 / 125

We should only execute this if enumext* is nested in the first level of enumext, for the rest of the cases the value of \g__enumext_item_number_int is already decreased.

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_check_ans_active:\ and\ \verb|_-enumext_check_ans_level:|)$

 $\verb|__enumext_check_ans_key_hook:|$

The function $_$ enumext_check_ans_key_hook: will *export* the status of the local variable $_$ enumext_check_ans_key_bool to the global variable $_$ enumext_check_ans_key_bool only if the key check-ans is active.

```
1850 \cs_new_protected:Nn \__enumext_check_ans_key_hook:
1851
       \bool_lazy_and:nnT
1852
         { \bool_if_p:N \l__enumext_check_ans_key_bool }
1853
         { \bool_if_p:N \g__enumext_standar_bool }
1854
1855
            \bool_gset_true:N \g__enumext_check_ans_key_bool
         }
       \bool_lazy_and:nnT
         { \bool_if_p:N \l__enumext_check_ans_key_bool }
         { \bool_if_p:N \g__enumext_starred_bool }
1861
            \bool_gset_true:N \g__enumext_check_ans_key_bool
1862
1863
     }
1864
```

(End of definition for __enumext_check_ans_key_hook:.)

__enumext_item_answer_diff:

The function __enumext_item_answer_diff: will set the value of the variable \g__enumext_item_-answer_diff_int which is used by the functions __enumext_check_ans_show: for the key saveans and by the function __enumext_check_ans_log: by the internal "check answer" mechanism. This function will be passed to the function __enumext_execute_after_env:.

```
1865 \cs_new_protected:Nn \__enumext_item_answer_diff:
1866 {
1867 \int_gset:Nn \g__enumext_item_answer_diff_int
1868 {
1869 \int_sign:n { \g__enumext_item_number_int - \g__enumext_item_anskey_int }
1870 }
1871 }
```

(End of definition for $_$ enumext_item_answer_diff:.)

__enumext_check_ans_show:
 __enumext_check_ans_msg_less:
 _enumext_check_ans_msg_same_ok:
 _enumext_check_ans_msg_greater:

The function __enumext_check_ans_show: will be executed within the function __enumext_-execute_after_env: when the key check-ans is active, that is, when \g__enumext_check_ans_-key_bool is "true" and will return the appropriate message according to the value of \g__enumext_-item_answer_diff_int set by the function __enumext_item_answer_diff:.

```
_{^{1872}} \cs_new_protected:Nn \__enumext_check_ans_show: _{^{1873}} { $_{\odot}2024$ by Pablo González L
```

```
\int_case:nn { \g__enumext_item_answer_diff_int }
           { -1 }{ \ enumext check ans msg less:
1876
             0 }{ \__enumext_check_ans_msg_same_ok: }
             1 }{ \__enumext_check_ans_msg_greater: }
1878
1879
    }
1880
   \cs_new_protected:Nn \__enumext_check_ans_msg_less:
       \msg_warning:nneee { enumext } { item-less-answer } { \g__enumext_store_name_tl }
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
    }
1885
   \cs_new_protected:Nn \__enumext_check_ans_msg_same_ok:
1886
    {
1887
       \msg_term:nneee { enumext } { items-same-answer } { \g__enumext_store_name_tl }
1888
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
1889
1890
   \cs_new_protected:Nn \__enumext_check_ans_msg_greater:
1891
    {
1892
       \msg_warning:nneee { enumext } { item-greater-answer } { \g__enumext_store_name_tl }
1893
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
```

(End of definition for __enumext_check_ans_show: and others.)

__enumext_check_ans_log:
_enumext_check_ans_log_msg_less:
_enumext_check_ans_log_msg_same_ok:
_enumext_check_ans_log_msg_greater:

The function __enumext_check_ans_log: will be executed within the function __enumext_-execute_after_env: when the key check-ans is not active, that is, when \g__enumext_check_-ans_key_bool is "false" and write in the log the appropriate message according to the value of \g__enumext_item_answer_diff_int set by the function __enumext_item_answer_diff:.

```
1896 \cs_new_protected:Nn \__enumext_check_ans_log:
1897
     {
       \int_case:nn { \g__enumext_item_answer_diff_int }
1898
1899
         {
           { -1 }{ \__enumext_check_ans_log_msg_less:
1900
              0 }{ \__enumext_check_ans_log_msg_same_ok: }
1901
              1 }{ \__enumext_check_ans_log_msg_greater: }
1902
1903
     }
1904
   \cs_new_protected:Nn \__enumext_check_ans_log_msg_less:
1906
       \msg_log:nneee { enumext } { item-less-answer } { \g__enumext_store_name_tl }
1907
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
1908
     }
1909
\cs_new_protected:Nn \__enumext_check_ans_log_msg_same_ok:
1911
       \msg_log:nneee { enumext } { items-same-answer } { \g__enumext_store_name_tl }
1912
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
1913
     }
1915 \cs_new_protected:Nn \__enumext_check_ans_log_msg_greater:
1916
       \msg_log:nneee { enumext } { item-greater-answer } { \g_enumext_store_name_tl }
1917
1918
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
1919
```

(End of definition for $_$ enumext_check_ans_log: and others.)

11.22.6 Writing .log and executing the check-ans key

__enumext_execute_after_env:

The __enumext_execute_after_env: function will first return the appropriate message for the end of the environment in which the save-ans key is being executed, then call the __enumext_item_-answer_diff: function and then will write the values of the global variables used to the .log file. If the key check-ans is active it will execute the function __enumext_check_ans_show: and show the result in the terminal, otherwise it will execute the function __enumext_check_ans_log: and write the results in the .log file will finally execute the function __enumext_reset_global_vars: returning the used variables to their original state. As this function is passed to the function __enumext_after_env:nn for the environments enumext and enumext* we must make sure that we are not nested at any level.

(End of definition for $\ensuremath{\verb|}_$ enumext_execute_after_env:.)

11.22.7 Check for \item* and \anspic* commands

__enumext_check_starred_cmd:n

The function __enumext_check_starred_cmd:n performs an extra check for the keyans, keyans* and keyanspic environments. Unlike the check executed by check-ans key this one is not controlled by any key, it is intended to prevent the forgetting of \item* or \anspic* in these environments.

```
\cs_new_protected:Npn \__enumext_check_starred_cmd:n #1
       \int_compare:nNnT
         { \g__enumext_check_starred_cmd_int } = { 0 }
         {
           \msg warning:nnnV
             { enumext } { missing-starred }{ #1 } \l__enumext_check_start_line_env_tl
1945
1946
       \int_compare:nNnT
1947
         { \g__enumext_check_starred_cmd_int } > { 1 }
1948
         {
1949
           \msg_warning:nnnV
1950
             { enumext } { many-starred }{ #1 } \l__enumext_check_start_line_env_tl
1951
       \int_gzero:N \g__enumext_check_starred_cmd_int
       \tl_clear:N \l__enumext_check_start_line_env_tl
     }
1955
```

(End of definition for $_$ enumext_check_starred_cmd:n.)

11.23 Keys and functions associated with storage

wrap-ans We add the keys wrap-ans, wrap-opt, save-sep, mark-ans, mark-pos, show-ans, show-pos, mark-wrap-opt ref and save-ref related to the "storage system" and internal mechanism of "label and ref" only at the first level of enumext and enumext*.

```
save-sep
mark-ans
          1956 \cs_set_protected:Npn \__enumext_tmp:n #1
mark-pos
show-ans
                 \keys_define:nn { enumext / #1 }
          1958
mark-ref
                     wrap-ans
                                .cs_set_protected:Np = \__enumext_anskey_wrapper:n ##1,
save-ref
                     wrap-ans
                                .initial:n = \fbox{##1},
                     wrap-ans
                                .value_required:n = true,
                     wrap-opt
                                .cs_set_protected:Np = \__enumext_keyans_wrapper_opt:n ##1,
                                .initial:n = [{##1}],
                     wrap-opt
                                .value required:n = true.
          1965
                     wrap-opt
                                .tl_set:N = \l__enumext_store_keyans_item_opt_sep_tl,
          1966
                     save-sep
                                .initial:n = {, ~ },
                     save-sep
          1967
                     save-sep
                                .value_required:n = true,
          1968
                                 .tl_set:N = \l__enumext_mark_answer_sym_tl,
                     mark-ans
                                 .initial:n = \textasteriskcentered,
                     mark-ans
          1970
                     mark-ans
                                 .value_required:n = true,
          1971
                                .choice:,
                     mark-pos
          1972
                     mark-pos / left   .code:n = \str_set:Nn \l__enumext_mark_position_str { l },
                     mark-pos / right .code:n = \str_set:Nn \l__enumext_mark_position_str { r },
                     mark-pos
                                .initial:n = right,
                                .value_required:n = true,
                     mark-pos
          1976
                                .bool_set:N = \l__enumext_show_answer_bool,
                     show-ans
          1977
                                .initial:n = false,
                     show-ans
          1978
                                .value_required:n = true,
          1979
                                .bool_set:N = \l__enumext_show_position_bool,
```

```
show-pos
                       .initial:n = false,
           show-pos
                      .value_required:n = true,
           mark-ref
                      .tl_set:N = \l__enumext_mark_ref_sym_tl,
1082
                      .initial:n = \textasteriskcentered,
           mark-ref
1984
           mark-ref
                      .value_required:n = true,
1985
           save-ref
                      .bool_set:N = \l__enumext_store_ref_key_bool,
1986
                      .initial:n = false,
           save-ref
1987
                      .value_required:n = true,
           save-ref
\clist_map_inline:nn { level-1, enumext* } { \__enumext_tmp:n {#1} }
(End of definition for wrap-ans and others.)
```

mark-pos For the keyans and keyans* environments we will only add the keys mark-pos, show-ans and show-show-ans pos.

show-pos

```
1992 \cs_set_protected:Npn \__enumext_tmp:n #1
1993
       \keys_define:nn { enumext / #1 }
1994
         {
           mark-pos .choice:,
           mark-pos / left .code:n = \str_set:Nn \l__enumext_mark_position_str { l },
           mark-pos / right .code:n = \str_set:Nn \l__enumext_mark_position_str { r },
1998
           mark-pos .initial:n = right,
1999
2000
           mark-pos .value_required:n = true,
           show-ans .bool_set:N = \l__enumext_show_answer_bool,
2001
           show-ans .initial:n = false,
2002
           show-ans .value_required:n = true,
2003
           show-pos .bool_set:N = \l__enumext_show_position_bool,
           show-pos .initial:n = false,
           show-pos .value_required:n = true,
2009 \clist_map_inline:nn { keyans, keyans* } { \__enumext_tmp:n {#1} }
```

 $(\mathit{End}\ of\ definition\ for\ mark-pos\ ,\ show-ans\ ,\ and\ show-pos.)$

11.23.1 Store optional arguments of the environments

The idea behind "storing" in the $\langle sequence \rangle$ is to have a copy of the structure of the environment in which the key save-ans is being executed so we must capture the optional arguments passed to the levels of the environment in which it is executed and "storing" them.

__enumext_store_active_keys:n
__enumext_store_active_keys_vii:n

The functions __enumext_store_active_keys:n and __enumext_store_active_keys_vii:n will be responsible for "storing" the $\langle keys \rangle$ filtered from the optional arguments of the environment in which the key save-ans is executed and the levels within this for the enumext and enumext* environments. We will execute this function only if the variable \l__enumext_store_save_key_X_bool is false, that is, the key store-key is not active, establishing the variable \l__enumext_store_save_key_X_tl with the filtered $\langle keys \rangle$.

```
2010 \cs_new_protected:Npn \__enumext_store_active_keys:n #1
2011
       \bool_if:cF { l__enumext_store_save_key_ \__enumext_level: _bool }
2012
2013
           \tl_clear:c { l__enumext_save_key_ \__enumext_level: _tl }
2014
           \tl_set:ce
2015
             { l__enumext_store_save_key_ \__enumext_level: _tl }
2016
              { \__enumext_filter_save_key:n {#1} }
2017
2018
2019
   \cs_new_protected:Npn \__enumext_store_active_keys_vii:n #1
2021
       \bool_if:NF \l__enumext_store_save_key_vii_bool
2022
           \tl_clear:N \l__enumext_store_save_key_vii_tl
2024
           \tl_set:Ne \l__enumext_store_save_key_vii_tl { \__enumext_filter_save_key:n {#1} }
2025
         }
2026
     }
2027
```

 $(\textit{End of definition for } \c enumert_store_active_keys:n \ and \c enumert_store_active_keys_vii:n.)$

11.23.2 Setting save-key key

Since this list structure will be stored in the $\langle sequence \rangle$ established by the save-ans key when executing \anskey, we will not be able to modify it. The best thing here is to have a key that allows you to modify the optional argument of the list stored in the $\langle sequence \rangle$.

save-key

The values set by this key passed in the optional arguments of the enumext and enumext* environments will override the values of the \l_enumext_store_save_key_X_tl variable set by the functions _enumext_store_active_keys:n and _enumext_store_active_keys_vii:n. Define the key save-key for all levels of enumext and enumext* environments.

(End of definition for save-key.)

__enumext_parse_save_key:n
_enumext_parse_save_key_vii:n

The functions __enumext_parse_save_key:n and __enumext_parse_save_key_vii:n will be responsible for storing the filtered $\langle keys \rangle$ in the variable \l__enumext_store_save_key_X_tl for enumext and enumext*.

```
2042 \cs_new_protected:Npn \__enumext_parse_save_key:n #1
    {
2043
       \bool_set_true:c { l__enumext_store_save_key_ \__enumext_level: _bool }
2044
       \tl_clear:c { l__enumext_save_key_ \__enumext_level: _tl }
2045
       \tl_set:ce
2046
         { l__enumext_store_save_key_ \__enumext_level: _tl }
         { \__enumext_filter_save_key:n {#1} }
2049
2050 \cs_new_protected:Npn \__enumext_parse_save_key_vii:n #1
2051
       \bool_set_true:N \l__enumext_store_save_key_vii_bool
2052
       \tl clear:N \l enumext store save key vii tl
2053
       \tl_set:Ne \l__enumext_store_save_key_vii_tl { \__enumext_filter_save_key:n {#1} }
2054
    }
2055
```

 $(\mathit{End of definition for} \ \verb|_-enumext_parse_save_key:n \ and \ \verb|_-enumext_parse_save_key_vii:n.))$

11.23.3 Internal functions to store optional arguments

__enumext_filter_save_key:n
__enumext_filter_save_key_key:n
__enumext_filter_save_key_pair:nn

The function __enumext_filter_save_key:n will be in charge of filtering the $\langle keys \rangle$ we want to *store* in $\langle sequence \rangle$ where $\{\#1\}$ represents the optional value passed to the environment.

The function __enumext_filter_save_key_key:n will be responsible for filtering the $\langle keys \rangle$ that are passed "without value" by excluding the resume, resume* and no-store keys.

The function $\ensuremath{\mbox{\mbox{$\setminus$}}}$ that are passed "with value" by excluding the series, resume, start, save-ans, save-ref, check-ans, show-ans, save-pos, wrap-ans, mark-ans, wrap-opt, save-sep, mark-ref, mini-env, mini-sep, miniright and miniright* keys.

```
2073 \cs_new:Npn \__enumext_filter_save_key_pair:nn #1#2
     {
       \str_case:nnF {#1}
2075
         {
2076
           { series
                         } {} { resume } {} { start
                                                          } {} { save-ans } {}
2077
           \{ \text{ save-ref } \} \{ \} \{ \text{ save-key } \} \{ \} \{ \text{ check-ans } \} \{ \} \}
2078
           { show-pos } {} { wrap-ans } {} { mark-ans } {} { wrap-opt } {}
           { save-sep } {} { mark-ref } {} { mini-env } {} { mini-sep } { }
           { miniright } {} { miniright* } {}
         }
         { , { \exp_not:n {\#1}} } = { \exp_not:n {\#2} } }
2083
2084
```

 $(End\ of\ definition\ for\ _enumext_filter_save_key:n\ ,\ _enumext_filter_save_key_key:n\ ,\ and\ __enumext_filter_save_key_pair:nn.)$

11.23.4 Function for storing content in prop list

__enumext_store_addto_prop:n
__enumext_store_addto_prop:V

The function $\ensuremath{\mbox{_enumext_store_addto_prop:n}}$ stores the content in $\ensuremath{\mbox{\mbox{$\langle$prop$ list$}\rangle$}}$ defined by save-ans key. The "stored content" is retrieved by means of the $\ensuremath{\mbox{$\rangle$}}$ defined by save-ans key.

The form in which the content is "stored" in the $\langle prop \ list \rangle$ is $\{\langle position \rangle\} \{\langle content \rangle\}$. This function is used by \anskey in enumext and enumext* environments, \item* in keyans and keyans* environments and \anspic in keyanspic environment.

```
compose the composition of the composition of
```

(End of definition for $_$ enumext_store_addto_prop:n.)

11.23.5 Function for storing content in sequence

with the same structure in which the command was executed.

__enumext_store_addto_seq:n
__enumext_store_addto_seq:v
__enumext_store_addto_seq:V

The function $_$ _enumext_store_addto_seq:n stores the content in $\langle sequence \rangle$ defined by save-ans key. This function is used by $\$ anskey in enumext, $\$ item* in keyans and $\$ anspic in keyanspic. The form in which the content is stored in $\langle sequence \rangle$ is in a internal enumext or enumext* environments

The "stored content" is retrieved by means of the \printkeyans command.

```
2094 \cs_new_protected:Npn \__enumext_store_addto_seq:n #1
2095 {
2096 \seq_gput_right:cn { g__enumext_ \l__enumext_store_name_tl _seq } { #1 }
2097 }
2098 \cs_generate_variant:Nn \__enumext_store_addto_seq:n { v, V }
```

 $(\mathit{End of definition for} \setminus __enumext_store_addto_seq:n.)$

11.23.6 Functions for storing the list structure in the sequence

__enumext_store_level_open:
\ enumext store level close;

The memorization structure of the list is handled by the functions __enumext_store_level_open: and __enumext_store_level_close: which are executed per level within the enumext environment.

__enumext_store_level_open_vii:
\ enumext store level close vii:

When nesting the <code>enumext*</code> environment in <code>enumext</code> starting right after <code>\item</code> (without material between them) there is a problem with the alignment of the labels with the baseline between the two environments. One way to get around this problem is to place <code>\mode_leave_vertical:</code> and then apply <code>\vspace</code> taking into account <code>\baselineskip</code>, the value of <code>\parsep</code> of the current level of <code>enumext</code> and the value of <code>\topsep</code> of the <code>enumext*</code> environment.

```
\cs_new_protected:Nn \__enumext_store_level_open_vii:
       \bool_if:NT \l__enumext_check_answers_bool
           \tl_if_empty:NTF \l__enumext_store_save_key_vii_tl
                  enumext store addto seg:n
                 {
                    \item \mode_leave_vertical:
2138
                      \vspace { -\skip_eval:n { \baselineskip + \parsep } }
                      \begin{enumext*}[before={\setlength{\topsep}{0pt}},]
                 }
             }
               \tl_put_left:Nn \l__enumext_store_save_key_vii_tl
                 {
                    \item \mode leave vertical:
                      \vspace { -\skip_eval:n { \baselineskip + \parsep } }
                      \begin{enumext*}[before={\setlength{\topsep}{0pt}},
2148
2149
               \tl_put_right:Nn \l__enumext_store_save_key_vii_tl
                 {
                 }
                 _enumext_store_addto_seq:V \l__enumext_store_save_key_vii_tl
         }
   \cs_new_protected:Nn \__enumext_store_level_close_vii:
2158
2159
       \bool_if:NT \l__enumext_check_answers_bool
2160
2161
              _enumext_store_addto_seq:n { \end{enumext*} }
         }
     }
```

(End of definition for __enumext_store_level_open_vii: and __enumext_store_level_close_vii:.)

11.23.7 Function for show marks and position

__enumext_print_keyans_box:NN __enumext_print_keyans_box:cc

The function __enumext_print_keyans_box:NN print a box in the left margin with \l__enumext_-mark_answer_sym_tl used by the wrap-ans, show-ans and show-pos keys. The function takes two arguments:

```
#1: \l__enumext_labelwidth_X_dim
#2: \l__enumext_labelsep_X_dim
```

11.24 The command \anskey and internal label and ref

(End of definition for $\ensuremath{\backslash}$ _enumext_print_keyans_box:NN.)

Since we will be "storing content" in a list environment within $\langle sequences \rangle$ and can (more or less) manage the options passed to each level, it is necessary that we have a little more control over \item when storing. The \anskey command will cover this point and give it very similar behaviour to that of \item in the enumext and enumext* environments.

\anskey We want the command to be executed as follows: $\anskey(\langle number \rangle)*[\langle key=val \rangle] \{\langle content \rangle\}$ so first we'll add the keys item-sym*, item-pos* and store-brk.

```
2179 \keys_define:nn { enumext / anskey }
2180 {
2181    item-sym* .tl_set:N = \l__enumext_store_item_symbol_tl,
2182    item-sym* .value_required:n = true,
2183    item-pos* .dim_set:N = \l__enumext_store_item_symbol_sep_dim,
2184    item-pos* .value_required:n = true,
2185    store-brk .bool_set:N = \l__enumext_store_columns_break_bool,
2186    store-brk .default:n = true,
2187    store-brk .value_forbidden:n = true,
```

This command \anskey will only be present when using the save-ans key in enumext and enumext* environments, otherwise it will return an error. If the check-ans key is active, increment \g_enumext_-count_item_with_ans_int, then call internal function _enumext_store_anskey_code:nnnn will "store content" in the $\langle sequence \rangle$ and in the $\langle prop list \rangle$.

```
NewDocumentCommand \anskey { d() s o +m }
    {
2190
       \bool if:NF \l enumext store active bool
2101
         {
           \msg_error:nnnn { enumext } { anskey-wrong-place }{ anskey }{ enumext }
2194
       \int_compare:nNnT { \l__enumext_keyans_level_int } = { 1 }
         {
           \msg_error:nnnn { enumext } { command-wrong-place }{ anskey }{ keyans }
       \int_compare:nNnT { \l__enumext_keyans_pic_level_int } = { 1 }
         {
           \msg_error:nnnn { enumext } { command-wrong-place }{ anskey }{ keyanspic }
         }
       \group begin:
         %\bool_if:NT \l__enumext_check_answers_bool
             \int_gincr:N \g__enumext_item_anskey_int
         \__enumext_store_anskey_code:nnnn {#1} {#2} {#3} {#4}
       \group_end:
    }
```

(End of definition for \anskey. This function is documented on page 11.)

__enumext_store_anskey_code:nnnn

The internal function __enumext_store_anskey_code:nnnn first we pass the command $\langle argument \rangle$ to the $\langle prop \ list \rangle$, then checks the state of the variable \l__enumext_store_ref_key_bool handled by the save-ref key and will call the function __enumext_store_internal_ref: for the internal "label and ref" system. Followed by this if the show-ans or show-pos keys are active we will show the "wrapped" $\langle argument \rangle$ passed to the command.

```
2211 \cs_new_protected:Npn \__enumext_store_anskey_code:nnnn #1 #2 #3 #4
2212 {
2213 \__enumext_store_addto_prop:n {#4}
2214 \bool_if:NT \l__enumext_store_ref_key_bool
2215 {
2216 \__enumext_store_internal_ref:
2217 }
2218 \__enumext_store_anskey_show_left:n { #4 }
```

Now we start processing the optional arguments passed to the command to build our \item in the variable \l__enumext_store_anskey_arg_tl which we will "store" in the $\langle sequence \rangle$. First we clear the variable \l__enumext_store_anskey_arg_tl and process $[\langle key=val \rangle]$, if the store-brk key is present and the command is running under enumext (not in the starred version) we will add \columnbreak and then \item.

Now we will check the $(\langle number \rangle)$ argument and add it to \l__enumext_store_anskey_arg_tl if the command is running under enumext* (starred version).

And now we will review the starred argument * together with the keys item-sym* and item-pos* and pass them to \l_enumext_store_anskey_arg_tl.

```
\bool_if:nTF {#2}
         {
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl { * }
           \tl_if_empty:NF \l__enumext_store_item_symbol_tl
             {
               \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
                 {
2248
                   [ \exp_not:V \l__enumext_store_item_symbol_tl ]
           \dim_compare:nT
             {
               \l
                  _enumext_store_item_symbol_sep_dim != \c_zero_dim
             3
               \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2258
                   [ \exp_not:V \l__enumext_store_item_symbol_sep_dim ]
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#4}
         }
         {
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#4}
```

Finally we check if the save-ref key is active along with the hyperref package load, if both conditions are met, it will create the \hyperlink and then store in \langle sequence \rangle.

```
\bool_lazy_and:nnT
```

```
{ \bool_if_p:N \l__enumext_store_ref_key_bool }

{ \bool_if_p:N \l__enumext_hyperref_bool }

{

**tl_put_right:Ne \l__enumext_store_anskey_arg_tl

**tl_put_right:Ne \l_enumext_store_anskey_arg_tl

**tl_put_right:Ne \l_
```

(End of definition for __enumext_store_anskey_code:nnnn.)

__enumext_store_internal_ref:

The function __enumext_store_internal_ref: handles the internal "label and ref" system used by the save-ref and mark-ref keys for \anskey will allow to execute \ref{ $\langle store\ name: position \rangle$ } and will return 1.(a).i.A.

First we will remove the dots "." from the current $\langle labels \rangle$, we do not want to get double dots in our references, then we will place this in the variable \l_enumext_newlabel_arg_two_tl.

```
cs_new_protected:Nn \__enumext_store_internal_ref:
{
cs_set_protected:Npn \__enumext_tmp:n ##1
{
    \tl_set_eq:cc { l__enumext_label_copy_##1_tl } { l__enumext_label_##1_tl }

    \tl_reverse:c { l__enumext_label_copy_##1_tl }

    \tl_remove_once:cn { l__enumext_label_copy_##1_tl } { . }

    \tl_reverse:c { l__enumext_label_copy_##1_tl } { . }

    \tl_reverse:c { l__enumext_label_copy_##1_tl }

    \tl_reverse:c { l__enumext_label_copy_##1_tl }

    \tl_reverse:c { l__enumext_label_copy_##1_tl }

    \tl_cist_map_inline:nn { i, ii, iii, iv, vii } { \__enumext_tmp:n {##1} }

    \cs_set:Npn \__enumext_tmp:n ##1

    \tl_use:c { l__enumext_label_copy_ \int_to_roman:n {##1} _tl } }
}
```

Here we need to analyse the cases where the environment is started with enumext* and if \anskey is running alone in it or if it is running in a nested enumext environment within the starting environment.

```
\bool_lazy_all:nT
         {
2292
           { \bool_if_p:N \g__enumext_starred_bool }
2293
           { \int_compare_p:nNn { \l__enumext_level_int } = { \c_zero_int } }
         }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
             { \tl_use:N \l__enumext_label_copy_vii_tl }
         }
       \bool_lazy_all:nT
         {
           { \bool_if_p:N \l__enumext_standar_bool }
2302
           { \bool_if_p:N \g__enumext_starred_bool }
2303
           { \int_compare_p:nNn { \l__enumext_level_int } > { \c_zero_int } }
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
               \tl_use:N \l__enumext_label_copy_vii_tl
               \int_step_function:nnN { 1 } { \l__enumext_level_int } \__enumext_tmp:n
```

If started with enumext and if \anskey is running alone in it or if it is running in a nested enumext* environment within the starting environment.

```
\text{bool_lazy_all:nT}

{
\text{ \bool_if_p:N \l_enumext_standar_bool \}}

{ \int_compare_p:nNn \ \l_enumext_level_int \} > \ \c_zero_int \} \}

{ \int_compare_p:nNn \ \l_enumext_level_h_int \} = \ \c_zero_int \} \}

{ \bool_not_p:n \ \l_enumext_starred_bool \} \}

}

{
\text{tl_put_right:Ne \l_enumext_newlabel_arg_two_tl}}

{
\text{tl_use:N \l_enumext_label_copy_i_tl}}

\int_step_function:nnN \{ 2 \} \ \l_enumext_level_int \} \_enumext_tmp:n

\end{array}
\text{tl_enumext_tmp:n}

\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_tmp:n}
\text{tl_enumext_t
```

```
}
        }
       \cs_set:Npn \__enumext_tmp:n ##1
         { \tl_use:c { l__enumext_label_copy_ \int_to_roman:n {##1} _tl } }
2328
       \bool_lazy_all:nT
         {
           { \bool_if_p:N \l__enumext_standar_bool }
           { \int_compare_p:nNn { \l__enumext_level_int } > { \c_zero_int } }
           { \bool_not_p:n { \g__enumext_starred_bool } }
           { \int_compare_p:nNn { \l__enumext_level_h_int } > { \c_zero_int } }
         }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
               \int_step_function:nnN { 1 } { \l__enumext_level_int } \__enumext_tmp:n
               . \tl_use:N \l__enumext_label_copy_vii_tl
2341
2342
```

Now we set the variable $\lower = 1 - enumext_newlabel_arg_one_tl$ which will contain $\{\langle store\ name: position \rangle\}$.

Now execute the function $_$ enumext_newlabel:nn and save the result in the variable $\l_$ enumext_store_write_aux_file_tl and finally we write in the .aux file.

```
table to the proof of the
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_store_internal_ref:.)$

__enumext_store_anskey_show_wrap:n

The function $\ensuremath{\mbox{\mbox{$\setminus$}}}$ enumext_store_anskey_show_wrap:n "wraps" the $\ensuremath{\mbox{$\langle$}}$ argument $\ensuremath{\mbox{$\rangle$}}$ passed to $\ensuremath{\mbox{$\setminus$}}$ when using the wrap-ans key.

(End of definition for $_$ enumext_store_anskey_show_wrap:n.)

__enumext_store_anskey_show_left:n

The function __enumext_store_anskey_show_left:n will show the "mark" defined by the markans key or the "position" of the content stored in the $\langle prop\ list \rangle$ when using the show-pos key on the left margin next to the "wraps" $\langle argument \rangle$ passed to \anskey on the right side when using the show-anskey.

```
2368 \cs_new_protected:Npn \__enumext_store_anskey_show_left:n #1
2369 {
2370    \bool_if:NT \l__enumext_show_answer_bool
2371    {
2372         \__enumext_store_anskey_show_wrap:n { #1 }
2373    }
2374    \bool_if:NT \l__enumext_show_position_bool
2375    {
2376         \tl_set:Ne \l__enumext_mark_answer_sym_tl
2377    {
```

```
\text{\group_begin:}
\text{\group_begin:}
\text{\group_not:N \normalfont}
\text{\group_not:N \footnotesize [\int_eval:n]
\text{\group_count:c { g__enumext_ \l_enumext_store_name_tl _prop }
\text{\group_count:c { g__enumext_ \l_enumext_store_name_tl _prop }
\text{\group_ass}
\text{\group_end:}
\text{\group_end:}
\text{\group_end:}
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_end:}
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_end:}
\text{\group_enumext_store_anskey_show_wrap:n { #1 }
\text{\group_enumext_store_
```

 $(\textit{End of definition for } \verb|_-enumext_store_anskey_show_left:n.)$

11.25 Common functions for keyans, keyans* and keyanspic

11.25.1 Storing content in prop list

__enumext_keyans_addto_prop:n

The function __enumext_keyans_addto_prop:n will pass the contents of the current $\langle label \rangle$ \l__enumext_label_v_tl for the keyans environment and the current $\langle label \rangle$ \l__enumext_label_vi_tl for the keyanspic environment when using \item* and \anspic*, followed by the contents of the optional argument of both commands to the \l__enumext_store_keyans_label_tl variable, which will be passed to the $\langle prop \ list \rangle$ defined by the save-ans key using the __enumext_store_addto_prop:V.

```
\cs_new_protected:Npn \__enumext_keyans_addto_prop:n #1
     {
2391
       \tl_clear:N \l__enumext_store_keyans_label_tl
       \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
         {
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \l__enumext_label_vi_tl }
         }
2396
         {
2397
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \l__enumext_label_v_tl }
2398
2399
       \tl_if_novalue:nF { #1 }
         {
           % Set save-sep
2402
           \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
               \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \l__enumext_store_keyans_item_op
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl { #1 }
2407
2408
         _enumext_store_addto_prop:V \l__enumext_store_keyans_label_tl
2409
     }
2410
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_keyans_addto_prop:n.)$

11.25.2 The save-ref key for keyans, keyans* and keyanspic

The internal "label and ref" system for the keyans, keyans* and keyanspic environments has slight differences with the one implemented for the \anskey command, basically because in this environments we are interested in the current $\langle label \rangle$. The mechanism defined here will allow to execute \ref{\store name: position}} and will return 1. (A).

__enumext_keyans_store_ref:
 __enumext_keyans_store_ref_aux_i:
 __enumext_keyans_store_ref_aux_ii:

The function __enumext_keyans_store_ref: handles the internal "label and ref" system used by the save-ref key for \item* and \anspic* commands. First we will create copies of the current $\langle labels \rangle$ and remove the dots "." from them, we do not want to get double dots in our references.

```
2411 \cs_new_protected:Nn \__enumext_keyans_store_ref:
2412
    {
       \bool_if:NT \l__enumext_store_ref_key_bool
2413
         {
2414
           \cs_set_protected:Npn \__enumext_tmp:n ##1
2415
             {
               \tl_set_eq:cc { l__enumext_label_copy_##1_tl } { l__enumext_label_##1_tl }
2417
               \tl_reverse:c { l__enumext_label_copy_##1_tl }
2418
               \tl_remove_once:cn { l__enumext_label_copy_##1_tl } { . }
               \tl_reverse:c { l__enumext_label_copy_##1_tl }
           \clist_map_inline:nn { i, v, vi, vii, viii } { \__enumext_tmp:n {##1} }
           \__enumext_keyans_store_ref_aux_i:
```

```
424 }
```

The auxiliary function __enumext_keyans_store_ref_aux_i: set the variable \l__enumext_newlabel_arg_one_tl which will contain $\{\langle store\ name: position \rangle\}$ analyzing whether the environment in which they are executed is enumext* or enumext.

```
\cs_new_protected:Nn \__enumext_keyans_store_ref_aux_i:
2427
       \bool_if:NT \g__enumext_starred_bool
           \tl_set_eq:NN \l__enumext_label_copy_i_tl \l__enumext_label_copy_vii_tl
       \int_compare:nNnT { \l__enumext_keyans_pic_level_int } = { 1 }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2434
             { \l__enumext_label_copy_i_tl . \l__enumext_label_copy_vi_tl }
       \int_compare:nNnT { \l__enumext_keyans_level_int } = { 1 }
2437
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
             { \l__enumext_label_copy_i_tl . \l__enumext_label_copy_v_tl }
       \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
             { \l__enumext_label_copy_i_tl . \l__enumext_label_copy_viii_tl }
2446
       \tl_put_right:Ne \l__enumext_newlabel_arg_one_tl
2447
           \l__enumext_store_name_tl \c_colon_str
           \int_eval:n { \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } }
         }
        \__enumext_keyans_store_ref_aux_ii:
2453
```

Now auxiliary function $_$ enumext_keyans_store_ref_aux_ii: save the result in the variable $_$ enumext_store_write_aux_file_tl and finally we write in the .aux file.

 $(End of definition for \verb|\| enumext| keyans| store| ref| aux_i:, and \verb|\| enumext| keyans| store| ref| aux_i:, and \verb|\| enumext| keyans| store| ref| aux_i:.)$

11.25.3 Storing content in sequence

__enumext_keyans_addto_seq:n
__enumext_keyans_addto_seq_link:

The function __enumext_keyans_addto_seq:n will pass the contents of the current $\langle label \rangle$ \l_-enumext_label_v_tl for the keyans environment and the \l_enumext_label_vi_tl for the keyanspic environment when using \item* and \anspic*, followed by the $\langle contents \rangle$ of the optional argument of both commands to the \l_enumext_store_keyans_label_tl variable to the sequence defined by the save-ans key.

```
2464 \cs_new_protected:Npn \__enumext_keyans_addto_seq:n #1
2465
    {
2466
       \tl_clear:N \l__enumext_store_keyans_label_tl
       \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
2467
         {
2468
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \item \l__enumext_label_vi_tl }
2469
         }
2470
         {
2471
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \item \l__enumext_label_v_tl }
2472
       \tl_if_novalue:nF { #1 }
           \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
             {
```

Checks if the save-ref key is active along with the hyperref package load, if both conditions are met, it will create the \hyperlink and then store using the __enumext_store_addto_seq:V function. Finally, copy the contents of the variable \l__enumext_store_keyans_label_tl into the global variable \g__enumext_check_ans_item_tl to be used by the function __enumext_check_starred_cmd:n and increment the value of the integer variable \g__enumext_item_anskey_int handled by the check-anskey.

```
2487 \cs_new_protected:Nn \__enumext_keyans_addto_seq_link:
    {
2488
       \bool_lazy_and:nnT
2489
         { \bool_if_p:N \l__enumext_store_ref_key_bool }
2490
         { \bool_if_p:N \l__enumext_hyperref_bool }
2491
           \tl_put_right:Ne \l__enumext_store_keyans_label_tl
2493
               \hfill \exp_not:N \hyperlink
                    \exp_not:V \l__enumext_newlabel_arg_one_tl
                  { \exp_not:V \l__enumext_mark_ref_sym_tl }
             }
         }
2501
       \__enumext_store_addto_seq:V \l__enumext_store_keyans_label_tl
       \bool_if:NT \l__enumext_check_answers_bool
2503
           \int_gincr:N \g__enumext_item_anskey_int
         }
```

 $(\textit{End of definition for } \verb|\|_enumext_keyans_addto_seq:n | and \verb|\|_enumext_keyans_addto_seq.link:|)$

11.25.4 The show-ans and show-pos keys for keyans and keyanspic

The code is very similar to the \anskey code, but, if I change the order of the operations the counter off $\langle label \rangle$ are incorrect.

__enumext_keyans_show_left:n
__enumext_keyans_show_ans:
__enumext_keyans_show_pos:
__enumext_keyans_show_item_opt:

Common function to show *starred commands* \item* and $\langle position \rangle$ of stored content in $\langle prop \ list \rangle$ for keyans and keyanspic. Need add 1 to \g__enumext_ $\langle store \ name \rangle$ _prop for show-pos key.

```
2508 \cs_new_protected:Npn \__enumext_keyans_show_left:n #1
       \tl_if_novalue:nF { #1 }
            \tl_set:Ne \l__enumext_keyans_item_opt_tl { #1 }
        \bool_if:NT \l__enumext_show_answer_bool
2515
             \__enumext_keyans_show_ans:
2516
        \bool_if:NT \l__enumext_show_position_bool
               _enumext_keyans_show_pos:
   \cs_new_protected:Nn \__enumext_keyans_show_item_opt:
        \tl_if_empty:NF \l__enumext_keyans_item_opt_tl
          {
2526
            \bool_lazy_or:nnT
              { \bool_if_p:N \l__enumext_show_answer_bool }
              { \bool_if_p:N \l__enumext_show_position_bool }
              {
                   _enumext_keyans_wrapper_opt:n {            <mark>\l__enumext_keyans_item_opt_tl</mark>        }             \c_space_tl
©2024 by Pablo González L
                                                                                                        74/125
```

```
}
     }
   \cs_new_protected:Nn \__enumext_keyans_show_ans:
2536
       \tl_put_left:Nn \l__enumext_label_v_tl
2537
              _enumext_print_keyans_box:NN
             \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
2541
     }
2542
   \cs_new_protected:Nn \__enumext_keyans_show_pos:
       \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
           \tl_set:Ne \l__enumext_mark_answer_sym_tl
             {
                \group_begin:
2549
                \exp_not:N \normalfont
                \exp_not:N \footnotesize [ \int_eval:n
                    \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
                 }
                 1
                \group_end:
2557
         }
2558
         {
           \tl_set:Ne \l__enumext_mark_answer_sym_tl
2560
             {
2561
                \group_begin:
                \exp_not:N \normalfont
               \exp_not:N \footnotesize [ \int_eval:n
                    \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } + 1
                 }
2568
                \group_end:
2569
       \tl_put_left:Nn \l__enumext_label_v_tl
              _enumext_print_keyans_box:NN
             \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
2575
         }
     }
```

($End\ of\ definition\ for\ _enumext_keyans_show_left:n\ and\ others.$)

Setting item-sym* and item-pos* keys

In order to have a cleaner implementation of \item* it is best to define a couple of keys that allow us to control and set by default the $\langle symbol \rangle$ and its $\langle offset \rangle$.

```
Define and set item-sym* and item-pos* keys for enumext and enumext*.
item-sym*
item-pos*
            2578 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
            2579
                   \keys_define:nn { enumext / #1 }
                       item-sym* .tl_set:c = { l__enumext_item_symbol_#2_tl },
                       item-sym* .value_required:n = true,
            2583
                       item-sym* .initial:n = {$\star$},
            2584
                       item-pos* .dim_set:c = { l__enumext_item_symbol_sep_#2_dim },
            2585
                        item-pos* .value_required:n = true,
            2586
                     }
            2587
            2588
            _{2589} \clist_map_inline:nn
                   {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {enumext*}{vii}
                 { \__enumext_tmp:nn #1 }
           (End of definition for item-sym* and item-pos*.)
           ©2024 by Pablo González L
```

11.27 Redefining \footnote command

__enumext_footnotetext:nn
__enumext_renew_footnote:
\ enumext print footnote:

To keep the correct numbering of \footnote and to make it work correctly with the mini-env key and in the enumext* and keyans* environments, it is necessary to redefine the command. This implementation is adapted from the answer given by Clea F. Rees (@cfr) in footnotes in boxes compatible with hyperref.

```
\cs_new_protected:Nn \__enumext_footnotetext:nn
      \footnotetext[#1]{#2}
2596
    }
2597
  \cs_new_protected:Nn \__enumext_renew_footnote:
2598
       \seq_gclear:N \g__enumext_footnote_arg_seq
      \seq_gclear:N \g__enumext_footnote_int_seq
      \RenewDocumentCommand \footnote { o +m }
          \tl_if_novalue:nTF {##1}
            {
2605
              \stepcounter{footnote}
2606
              \int_gset_eq:Nc \g__enumext_footnote_int { c@footnote }
2607
              \int_gset:Nn \g__enumext_footnote_int { ##1 }
          \footnotemark [ \g__enumext_footnote_int ]
          \seq_gput_right:Nn \g__enumext_footnote_arg_seq { ##2 }
          2614
       }
2615
2616
  \cs_new_protected:Nn \__enumext_print_footnote:
2617
2618
      \seq_if_empty:NF \g__enumext_footnote_int_seq
2619
        {
2620
          \seq_map_pairwise_function:NNN
            \g__enumext_footnote_int_seq
            \g__enumext_footnote_arg_seq
            \__enumext_footnotetext:nn
        }
2625
    }
2626
```

 $(\textit{End of definition for } \\ _\texttt{enumext_footnoteext:nn}, \\ \\ _\texttt{enumext_renew_footnote:}, \\ \textit{and } \\ \\ _\texttt{enumext_print_footnotee:})$

11.28 Redefining \item command

Redefining the \item command is not as simple as I thought. This command works in conjunction with the \makelabel command so I have to redefine both of them, in addition to this, we will have to use a couple of global variables to pass the values from one command to the other.

11.28.1 The \item command in enumext

__enumext_default_item:n

The $\idetical item and $\idetical item [\langle custom \rangle]$ commands work in the usual way on enumext.$

First we will see if the optional argument is present, if it is NOT present we will check the state of the variable \l__enumext_check_ans_key_bool set by the key check-ans, set the boolean variable \l__enumext_wrap_label_X_bool to "true" and execute __enumext_item_std:w.

The boolean variable \l__enumext_wrap_label_X_bool is used by the function __enumext_make_-label: (§11.29).

```
\cs_new_protected:Npn \__enumext_default_item:n #1
       \tl_if_novalue:nTF {#1}
         {
           \bool_if:NT \l__enumext_check_answers_bool
2631
             {
2632
               \int_gincr:N \g__enumext_item_number_int
2633
2634
           \bool_set_true:c { l__enumext_wrap_label_ \__enumext_level: _bool }
2635
           \__enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ \__enumext_level: _tl }
2636
         }
2637
         {
           \bool_set_eq:cc
             { l__enumext_wrap_label_ \__enumext_level: _bool }
```

76 / 125

(End of definition for $_$ enumext_default_item:n.)

__enumext_starred_item:nn

The $\identified item^*, \iden^*[\langle symbol \rangle] \ and \iden^*[\langle symbol \rangle] \ [\langle offset \rangle] \ works like the numbered \identified, but placing a <math>[\langle symbol \rangle]$ to the "left" of the $\langle label \rangle$ separated from it by the value set by the labelsep key and can be offset using the second optional argument $[\langle offset \rangle]$.

```
#1: \l_enumext_item_symbol_X_tl
#2: \l_enumext_item_symbol_sep_X_dim
```

First we will make a copy of $\l_=\text{enumext_item_symbol_X_tl}$ which is set by the key item-sym* or passed as optional argument in the global variable $\g_=\text{enumext_item_symbol_tl}$, followed by setting the variable $\l_=\text{enumext_item_symbol_sep_X_dim set}$ by the key item*-sep or by the second optional argument.

Then we will see the state of the variable $\l_enumext_check_ans_key_bool$ set by the key check-ans, set the boolean variable $\l_enumext_wrap_label_X_bool$ to "true" and execute $\l_enumext_item_std:w$.

In this function the optional argument of __enumext_item_std:w is omitted, we only want it to be numbered.

The boolean variable $\lower = 1.2$ and the vars $\lower = 1.2$ and the

```
2645 \cs_new_protected:Npn \__enumext_starred_item:nn #1 #2
      \tl_if_novalue:nF {#1}
2647
2648
          \tl_set:cn { l__enumext_item_symbol_ \__enumext_level: _tl } {#1}
      \tl_if_novalue:nTF {#2}
2653
        {
          \dim_set_eq:cc
2654
            { l__enumext_item_symbol_sep_ \__enumext_level: _dim }
2655
            { l__enumext_labelsep_ \__enumext_level: _dim }
2656
        }
2657
        {
2658
          \dim_set:cn { l__enumext_item_symbol_sep_ \__enumext_level: _dim } {#2}
        }
      \bool_if:NT \l__enumext_check_answers_bool
2662
          \int_gincr:N \g__enumext_item_number_int
2662
2664
      \bool_set_true:c { l__enumext_wrap_label_ \__enumext_level: _bool }
2665
      \__enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ \__enumext_level: _tl }
2666
```

__enumext_redefine_item:

The function __enumext_redefine_item: will redefine the \item command in the enumext environment for the internal mechanism of check-answers for check-ans key and adding the starred \item* version.

This function is passed to __enumext_list_arg_two_X: which is used in the definition of the enumext environment (§11.30.2).

 $(End\ of\ definition\ for\ \verb|_-enumext_redefine_item:.)$

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_starred_item:nn.)$

```
©2024 by Pablo González L
```

11.28.2 The \item command in keyans

The $\idesigned \mbox{"item*} [\langle content \rangle] \mbox{ commands } store \mbox{ the current } \langle label \rangle \mbox{ next to the } [\langle content \rangle] \mbox{ if it is present in the } \langle sequence \rangle \mbox{ and } \langle prop \mbox{ list} \rangle \mbox{ defined by save-ans key.}$

__enumext_keyans_default_item:n

The function __enumext_keyans_default_item:n executes the original behavior of the \item.

(End of definition for __enumext_keyans_default_item:n.)

__enumext_keyans_starred_item:n

The function __enumext_keyans_starred_item:n which will make a temporary copy of the current $\langle label \rangle$, execute the show-ans or show-pos keys using the function __enumext_keyans_show_left:n and will display the contents of that item using the internal copy __enumext_item_std:w, this is necessary to prevent incrementing the current "counter" of the original $\langle label \rangle$.

Recover the original value of the current $\langle label \rangle$ and *store* it first in the $\langle prop \ list \rangle$ (including the optional argument), run the internal "label and ref" system if the save-ref key is active and finally *store* it in the $\langle sequence \rangle$.

```
\tl_set_eq:NN \l__enumext_label_v_tl \l__enumext_keyans_tmpa_tl
\_enumext_keyans_addto_prop:n { #1 }
\_enumext_keyans_store_ref:
\_enumext_keyans_addto_seq:n { #1 }
\int_gincr:N \g_enumext_check_starred_cmd_int
\]
```

(End of definition for $\label{lem:n.}$) = enumext_keyans_starred_item:n.)

__enumext_keyans_redefine_item:

The function __enumext_keyans_redefine_item: is responsible for adding the *starred* and *optional* argument by the __enumext_list_arg_two_v: function in the definition of the keyans environment. Here we need to use \peek_remove_spaces:n to prevent an unwanted space when using \item* in conjunction with the itemindent key.

This function is passed to $\ensuremath{\mbox{\mbox{$\setminus$}}}$ environment ($\S11.30.2$).

```
2703 \cs_new_protected:Nn \__enumext_keyans_redefine_item:
     {
2704
       \RenewDocumentCommand \item { s o }
           \bool_if:nTF {##1}
             {
                \peek_remove_spaces:n
                  {
                       _enumext_keyans_starred_item:n {##2}
                  }
             7
2714
                  _enumext_keyans_default_item:n {##2}
             }
         }
     }
```

(End of definition for \item* and __enumext_keyans_redefine_item:. This function is documented on page 12.)

11.29 Redefining \makelabel command

Redefine \makelabel for the keys align, font, wrap-label, wrap-label* and \item* for enumext and keyans environments.

11.29.1 Redefining \makelabel for enumext

 $(\mathit{End}\ of\ definition\ for\ \verb|__enumext_item_starred:.)$

__enumext_item_starred:

The function __enumext_item_starred: will be responsible for executing \item* for the enumext environment.

enumext make label:

The function __enumext_make_label: redefine \makelabel for the enumext environment.

This function is passed to __enumext_list_arg_two_X: which is used in the definition of the enumext environment (§11.30.2).

(End of definition for __enumext_make_label:.)

11.29.2 Redefining \makelabel for keyans

__enumext_keyans_make_label:

The function __enumext_keyans_make_label: redefine \makelabel for keyans environment.

This function is passed to __enumext_list_arg_two_v: which is used in the definition of the keyans environment (§11.30.2).

 $(\textit{End of definition for } \verb|_-enumext_keyans_make_label:.)$

11.30 Second argument of the lists

At this point of the code we have already programmed most the necessary tools to create a custom list environment, remember that the function __enumext_start_list:nn takes two arguments, the first one we have ready, the second one we will define for all the levels of the environment enumext and the environment keyans.

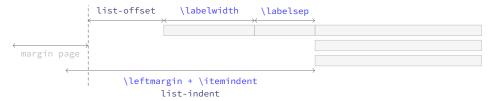


Figure 9: Representation of standard horizontal lengths in list environment.

11.30.1 Calculation of \leftmargin and \itemindent

Consider the figure 9 where the default margins (on the left) of a list are represented.

The idea is to have control over these margins so that our list does not overlap the left margin of the page. The *key* relationship is that the right edge of the \labelsep equals the right edge of the \itemindent, so that the left edge of the *label box* is at \left\text{leftmargin+\itemindent} minus \label\width+\labelsep. Thus, the handling of the margins by the package will be as shown in the figure 10.

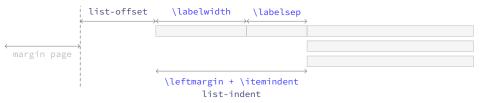


Figure 10: Representation of horizontal lengths concept in list in enumext.

Where the default values will look like in the figure 11.

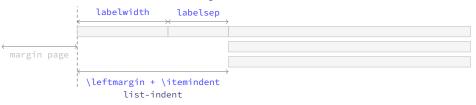


Figure 11: Default horizontal lengths in enumext.

__enumext_calc_hspace:NNNNNNN\ _enumext_calc_hspace:cccccc The function __enumext_calc_hspace: NNNNNNN takes seven arguments to be able to determine horizontal spaces for all list environment:

```
#1: \l__enumext_labelwidth_X_dim #2: \l__enumext_labelsep_X_dim
#3: \l__enumext_listoffset_X_dim #4: \l__enumext_leftmargin_tmp_X_dim
#5: \l__enumext_leftmargin_X_dim #6: \l__enumext_itemindent_X_dim
#7: \l__enumext_leftmargin_tmp_X_bool
```

And returns the "adjusted" values of \leftmargin and \itemindent.

This function is passed to __enumext_list_arg_two_X: which is used in the definition of the enumext and keyans environments (§11.30.2).

If no value has been passed to the labelwidth and labelsep keys we set the default values for \l_- enumext_leftmargin_tmp_X_dim.

```
\bool_if:nF #7 { \dim_set:Nn #4 { #1 + #2} }
```

We now analyze the cases and set the values for \leftmargin and \itemindent.

```
\dim_compare:nNnTF { #4 } < { \c_zero_dim }

2773 {

2774 \dim_set:Nn #6 { #1 + #2 - #4}

2775 \dim_set:Nn #5 { #1 + #2 + #3 - #6 }

2776 }
```

__enumext_list_arg_two_i:

__enumext_list_arg_two_ii: __enumext_list_arg_two_iii:

__enumext_list_arg_two_iv:

__enumext_list_arg_two_v:

__enumext_list_arg_two_vii: __enumext_list_arg_two_viii:

```
\dim_compare:nNnT { #4 } = { #1 + #2 }
             { \dim_set:Nn #6 { \c_zero_dim } }
           \dim_compare:nNnT { #4 } < { #1 + #2 }
             { \dim_set:Nn #6 { #1 + #2 - #4} }
2781
           \dim_compare:nNnT { #4 } > { #1 + #2 }
2782
2783
                \dim_set:Nn #6 { -#1 - #2 + #4}
                \dim_set:Nn #6 { #6*-1}
2785
           \dim_set:Nn #5 { #1 + #2 + #3 - #6 }
2789
2790 \cs_generate_variant:Nn \__enumext_calc_hspace:NNNNNNN { ccccccc }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_calc_hspace: \verb|NNNNNN|)$

11.30.2 Setting second argument of the lists

We will "not set" \leftmargini, \leftmarginii, \leftmarginiii or \leftmarginiv, in this case, we will directly set the parameters for vertical and horizontal list spacing per level.

```
\cs_set_protected:Npn \__enumext_tmp:n #1
    {
2792
       \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
2793
         {
           \__enumext_calc_hspace:cccccc
             { l__enumext_labelwidth_#1_dim } { l__enumext_labelsep_#1_dim }
             { l__enumext_listoffset_#1_dim } { l__enumext_leftmargin_tmp_#1_dim }
             { l__enumext_leftmargin_#1_dim } { l__enumext_itemindent_#1_dim }
             { l__enumext_leftmargin_tmp_#1_bool }
2799
           \clist_map_inline:nn
2800
             { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
2801
             { \dim_set_eq:cc {####1} { l__enumext_####1_#1_dim } }
2802
           \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
2803
             { \skip_set_eq:cc {####1} { l__enumext_####1_#1_skip } }
           \usecounter { enumX#1 }
           \setcounter { enumX#1 } { \int_eval:n { \int_use:c { l__enumext_start_#1_int } - 1 } }
           \str_if_eq:nnTF {#1} { v }
             {
               \ enumext keyans make label:
2810
               \__enumext_keyans_ref:
2811
               \__enumext_keyans_fake_item:
2812
               \bool_if:cT { l__enumext_show_length_#1_bool }
2813
                 {
                   \msg_term:nnnn { enumext } { list-lengths-not-nested } { v } { keyans }
                 }
             }
2818
               \__enumext_redefine_item:
2819
               \__enumext_make_label:
2820
               \__enumext_standar_ref:
2821
               \__enumext_fake_item:
2822
               \bool_if:cT { l__enumext_show_length_#1_bool }
2823
                   \msg_term:nnne { enumext } { list-lengths } {#1} { \int_use:N \l__enumext_level_i
                 }
             }
         }
2828
2830 \clist_map_inline:nn { i, ii, iii, iv, v } { \__enumext_tmp:n {#1} }
```

(End of definition for $_=$ enumext_list_arg_two_i: and others.)

For the horizontal environments enumext* and keyans* the implementation is similar, but, the value of \partopsep is always opt. At this point we will modify the parsep key to make it take the value of the itemsep key and later, in the environment definition, we will modify parindent to make it set the value of lisparindent and parsep to set the value of \parskip locally.

```
2831 \cs_set_protected:Npn \__enumext_tmp:n #1
       \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
©2024 by Pablo González L
```

81/125

```
\__enumext_calc_hspace:cccccc
             { l__enumext_labelwidth_#1_dim } { l__enumext_labelsep_#1_dim }
             { l__enumext_listoffset_#1_dim } { l__enumext_leftmargin_tmp_#1_dim }
             { l__enumext_leftmargin_#1_dim } { l__enumext_itemindent_#1_dim }
2838
             { l__enumext_leftmargin_tmp_#1_bool }
           \clist_map_inline:nn
             { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
2841
              { \dim_set_eq:cc {####1} { l__enumext_###1_#1_dim } }
           \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
2843
              { \skip_set_eq:cc {####1} { l__enumext_####1_#1_skip } }
           \skip_set_eq:Nc \parsep { l__enumext_itemsep_#1_skip }
           \skip_zero:N \partopsep
           \usecounter { enumX#1 }
2847
           \setcounter { enumX#1 } { \int_eval:n { \int_use:c { l__enumext_start_#1_int } - 1 } }
2848
           \__enumext_starred_ref:
2849
           \str_if_eq:nnTF {#1} { vii }
2850
             {
2851
                \__enumext_fake_item_vii:
2852
                \bool_if:cT { l__enumext_show_length_vii_bool }
2853
                  { \msg_term:nnnn { enumext } { list-lengths-not-nested } { vii } { enumext* } }
             }
2855
                \__enumext_fake_item_viii:
                \bool_if:cT { l__enumext_show_length_#1_bool }
2858
                  { \msg_term:nnnn { enumext } { list-lengths-not-nested } { #1 } { keyans* } }
2859
             }
2860
         }
2861
2863 \clist_map_inline:nn { vii, viii } { \__enumext_tmp:n {#1} }
(End of definition for \__enumext_list_arg_two_vii: and \__enumext_list_arg_two_viii:.)
```

11.31 The environment enumext

enumext We create the enumext environment based on list environment by levels.

```
2864 \NewDocumentEnvironment{enumext}{ 0{}} }
     {
2865
       \__enumext_safe_exec:
2866
       \__enumext_parse_keys:n {#1}
       \ enumext before list:
2868
       \__enumext_start_store_level:
2869
       \__enumext_start_list:nn
2870
         { \tl_use:c { l__enumext_label_ \__enumext_level: _tl } }
2871
2872
            \use:c { __enumext_list_arg_two_ \__enumext_level: : }
2873
            \__enumext_before_keys_exec:
         }
       \__enumext_after_args_exec:
2876
2877
     }
2878
       \__enumext_stop_list:
2879
       \__enumext_stop_store_level:
2880
       \__enumext_after_list:
2881
```

\ enumext safe exec:

The __enumext_safe_exec: function first execute the function __enumext_is_not_nested: which will set the variable \g__enumext_standar_bool to "true" if the environment is not nested in enumext*, we increment the variable \l__enumext_level_int for the nesting levels and set the \l__enumext_standar_bool variable to "true". Finally we set the variable \l__enumext_standar_first_bool to "true" only if the environment is not nested and we are at the "first level" of it using the function __enumext_is_on_first_level:.

```
2883 \cs_new_protected:Nn \__enumext_safe_exec:
2884 {
2885 \__enumext_is_not_nested:
2886 \int_incr:N \l__enumext_level_int
2887 \int_compare:nNnT { \l__enumext_level_int } > { 4 }
2888 { \msg_fatal:nn { enumext } { \list-too-deep } }
2889 \bool_set_true:N \l__enumext_standar_bool
2890 \__enumext_is_on_first_level:
2891 }

©2024 by Pablo González L
```

(End of definition for enumext. This function is documented on page 4.)

82 / 125

(End of definition for __enumext_safe_exec:.)

__enumext_parse_keys:n

The __enumext_parse_store_keys:n function will parse the $\langle keys \rangle$ passed to the optional environment argument enumext by levels only if present. First we clear the variable \l__enumext_series_str and then we check if we are at the first level, if so we process the $\langle keys \rangle$ and then execute the function __enumext_parse_series:n used by the key series, otherwise we will pass the $\langle keys \rangle$ to the inner levels of the environment and finally if the variable \l__enumext_store_active_bool established by the key save-ans is true we execute __enumext_parse_store_keys:n used by the key save-key.

(End of definition for $_$ enumext_parse_keys:n.)

__enumext_start_store_level:
__enumext_stop_store_level:

The __enumext_start_store_level: and __enumext_stop_store_level: functions activate the level saving mechanism for storage in \(\sequence \rangle \) of the \anskey command.

If enumext are nested in enumext* add __enumext_store_level_open: to preserve the stored structure.

```
2909 \cs_new_protected:Nn \__enumext_start_store_level:
       \bool_lazy_all:nT
2011
2912
            { \bool_if_p:N \l__enumext_store_active_bool }
2913
            { \bool_not_p:n { \l__enumext_keyans_env_bool } }
2914
            { \bool_not_p:n { \g__enumext_starred_bool } }
2915
2916
2917
            \int_compare:nNnT { \l__enumext_level_int } > { 1 }
                \bool_set_true:c { l__enumext_store_upper_level_ \__enumext_level: _bool }
                 \__enumext_store_level_open:
          }
        \bool_lazy_all:nT
2924
2925
          {
            { \bool_if_p:N \l__enumext_store_active_bool }
            { \bool_not_p:n { \l__enumext_keyans_env_bool } }
2927
            { \bool_if_p:N \g__enumext_starred_bool }
          }
          {
            \int_compare:nNnT { \l__enumext_level_int } > { 0 }
              {
                \bool_set_true:c { l__enumext_store_upper_level_ \__enumext_level: _bool }
2933
                 \__enumext_store_level_open:
2935
          }
2936
2937
   \cs_new_protected:Nn \__enumext_stop_store_level:
2938
     {
2939
        \bool_if:cT { l__enumext_store_upper_level_ \__enumext_level: _bool }
2940
             \__enumext_store_level_close:
          }
     }
2944
(End\ of\ definition\ for\ \_enumext\_start\_store\_level:\ and\ \_enumext\_stop\_store\_level:.)
```

__enumext_before_list:

The function __enumext_before_list: will add the vertical spacing on the environment if the above key is active next to the $\{\langle code \rangle\}$ defined by the before* key if it is active.

```
2945 \cs_new_protected:Nn \__enumext_before_list:
2946 {
2947 \__enumext_vspace_above:
2948 \__enumext_before_args_exec:
```

The function __enumext_check_ans_active: will handle the check answer mechanism, which will be activated with the check-ans key.

```
49 \__enumext_check_ans_active:
```

When the mini-env key is active it will set the value of the \l__enumext_minipage_right_X_dim to be the width of the __enumext_mini_env* environment on the "right side", using this value together with the value of the \l__enumext_minipage_hsep_X_dim set by the mini-sep key, the value of \l__enumext_minipage_left_X_dim will be set, which will be the width of __enumext_mini_env* environment on the "left side", always having a current \linewidth as maximum width between them.

The boolean variable \l__enumext_minipage_active_X_bool will be activated and the integer variable \g__enumext_minipage_stat_int used by the \miniright command will be incremented, then the function __enumext_mini_addvspace: is called and the __enumext_mini_env* environment on the "left side" will be initialized followed by the "vertical spacing" applied to preserve the "baseline" between the left and right side environments. After these actions, the function __enumext_multicols_start: is called to handle the multicols environment.

Here we use the plain TEX macro \nointerlineskip to prevent baseline "glue" being added between the next pair of boxes in a vertical list.

```
bool_set_true:c { l__enumext_minipage_active_ \__enumext_level: _bool }

int_gincr:N \g__enumext_minipage_stat_int

__enumext_mini_addvspace:

| \ointerlineskip\noindent
| \begin{__enumext_mini_env*}
| \dim_use:c { l__enumext_minipage_left_ \__enumext_level: _dim } }

| \__enumext_multicols_start:
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_before_list:.)$

__enumext_multicols_start:

The function __enumext_multicols_start: will start the multicols environment according to the value passed by the columns key, then set the default value for \columnsep when columns-sep=0pt and set the value of \multicolsep equal to zero and leave \columnseprule equal to zero for inner levels.

```
2968 \cs_new_protected:Nn \__enumext_multicols_start:
2969
       \int compare:nNnT
2970
         { \int_use:c { l__enumext_columns_ \__enumext_level: _int } } > { 1 }
2971
2972
           \dim compare:nNnT
2973
             { \dim_use:c { l__enumext_columns_sep_ \__enumext_level: _dim } } = { \c_zero_dim }
2974
2975
                \dim_set:cn { l__enumext_columns_sep_ \__enumext_level: _dim }
                    ( \dim_use:c { l__enumext_labelwidth_ \__enumext_level: _dim }
                      + \dim_use:c { l__enumext_labelsep_ \__enumext_level: _dim }
                    ) / \int_use:c { l__enumext_columns_ \__enumext_level: _int }
                     - \dim_use:c { l__enumext_listoffset_ \__enumext_level: _dim }
2981
                  }
2982
             }
2983
           \dim_set_eq:Nc \columnsep { l__enumext_columns_sep_ \__enumext_level: _dim }
           \skip_zero:N \multicolsep
           \int_compare:nNnT { \l__enumext_level_int } > { 1 }
                \dim_zero:N \columnseprule
             }
©2024 by Pablo González L
                                                                                                  84 / 125
```

We will calculate the *vertical spacing* settings for the multicols environment using the function __enumext_multi_addvspace:, apply our "*vertical adjust spacing*", then start the multicols environment.

__enumext_multicols_stop:

The function __enumext_multicols_stop: will stop the multicols environment. If the boolean variable \l__enumext_minipage_active_X_bool is false (not nested in __enumext_mini_env*) we will apply our "vertical adjust" spacing.

(End of definition for __enumext_multicols_stop:.)

(End of definition for $_=$ enumext_multicols_start:.)

__enumext_after_list:

The function __enumext_after_list: will will check the state of the boolean variable \l__enumext_minipage_active_X_bool, if it is "true" a small test will be executed to check if we have omitted the use of \miniright (the __enumext_mini_env* environment has not been closed), then close __enumext_mini_env* and add the adjusted vertical space \l__enumext_minipage_after_skip, otherwise we will close the multicols environment.

```
3010 \cs_new_protected:Nn \__enumext_after_list:
3011
       \bool_if:cTF { l__enumext_minipage_active_ \__enumext_level: _bool }
3012
3013
            \int_compare:nNnT { \g__enumext_minipage_stat_int } = { 1 }
                \msg_warning:nn { enumext } { missing-miniright }
                \miniright
3017
3018
            \verb|\int_gzero:N \  \  \| g_{\_enumext\_minipage\_stat\_int}|
3019
            \end{__enumext_mini_env*}
            \par\addvspace { \l__enumext_minipage_after_skip }
3021
          { \__enumext_multicols_stop: }
```

If the check-ans key is active, we set the boolean variable $g_enumext_check_ans_show_bool$ to true and copy the "store name" to the variable $g_enumext_store_name_tl$.

```
\__enumext_check_ans_key_hook:
```

Now apply the $\{\langle code \rangle\}$ handled by the after key together with the *vertical space* handled by the below key if they are present, set \l__enumext_standar_bool to false and save the *current value* of the counter for series, resume and resume* keys.

```
3025 \__enumext_after_stop_list:
3026 \__enumext_vspace_below:
3027 \bool_set_false:N \l__enumext_standar_bool
3028 \__enumext_resume_save_counter:
3029 }
```

(End of definition for $__$ enumext $_$ after $_$ list:.)

As we don't want our check to be executed check-ans by levels but on the complete list, we will take it out of the enumext environment using the "hook" function __enumext_after_env:nn.

```
3030 \__enumext_after_env:nn {enumext} { \__enumext_execute_after_env: }
```

11.32 The environment keyans

The environment keyans also based on lists. The main differences with the enumext environment are the nesting and the way the answers (choice) will be stored and checked, this environment is intended exclusively for "multiple choice questions".

Now we define the environment keyans also based on lists.

```
3031 \NewDocumentEnvironment{keyans}{ 0{} }
3032
        \__enumext_keyans_safe_exec:
3033
       \__enumext_keyans_parse_keys:n {#1}
3034
       \__enumext_before_list_v:
3035
       \__enumext_start_list:nn
          { \tl_use:N \l__enumext_label_v_tl }
3037
          {
              _enumext_list_arg_two_v:
            \__enumext_before_keys_exec_v:
3040
3041
       \__enumext_after_args_exec_v:
3042
     }
3043
3044
       \__enumext_check_starred_cmd:n { item }
3045
       \__enumext_stop_list:
3046
       \__enumext_after_list_v:
```

(End of definition for keyans. This function is documented on page 11.)

__enumext_keyans_safe_exec:

The keyans environment will only be available if the save-ans key is active and can only be used at the first level within the enumext environment. We do not want the environment to be nested, so we will set a maximum at this point. If the conditions are not met, an error message will be returned.

```
\cs_new_protected:Nn \__enumext_keyans_safe_exec:
     {
3050
        \bool_if:NF \l__enumext_store_active_bool
3051
          {
3052
            \msg_error:nnnn { enumext } { wrong-place }{ keyans }{ save-ans }
3053
          }
        \int_incr:N \l__enumext_keyans_level_int
3055
        \bool_set_true:N \l__enumext_keyans_env_bool
        \__enumext_keyans_save_start_line:
        % Set false for interfering with enumext nested in keyans (yes, its possible and crayze)
3058
        \bool_set_false:N \l__enumext_store_active_bool
3059
        \int_compare:nNnT { \l__enumext_keyans_level_int } > { 1 }
3060
          {
3061
            \msg_error:nn { enumext } { keyans-nested }
        \int_compare:nNnT { \l__enumext_level_int } > { 1 }
            \msg_error:nn { enumext } { keyans-wrong-level }
          }
     }
3068
(End of definition for \__enumext_keyans_safe_exec:.)
Parse [\langle key = val \rangle] for keyans environment.
3069 \cs_new_protected:Npn \__enumext_keyans_parse_keys:n #1
```

_enumext_keyans_parse_keys:n

```
{
        \keys_set:nn { enumext / keyans } {#1}
3071
     }
3072
```

 $(\mathit{End}\ of\ definition\ for\ \verb|__enumext_keyans_parse_keys:n.)$

__enumext_before_list_v:

The function __enumext_before_list_v: will add the vertical spacing above the environment if the above key is active next to the $\langle code \rangle$ defined by the before key if it is active.

```
3073 \cs_new_protected:Nn \__enumext_before_list_v:
    {
3074
          enumext vspace above v:
3075
       \__enumext_before_args_exec_v:
3076
```

When the mini-env key is active it will set the value of the \l__enumext_minipage_right_v_dim to be the width of the __enumext_mini_env* environment on the left side, using this value together with the value of the \l__enumext_minipage_hsep_v_dim set by the mini-sep key, the value of \l__enumext_minipage_left_v_dim will be set, which will be the width of __enumextt_mini_env* environment on the right side, always having \linewidth as the maximum width between them.

```
dim_compare:nNnT { \l_enumext_minipage_right_v_dim } > { \c_zero_dim }

{
dim_set:Nn \l_enumext_minipage_left_v_dim
}

{
    \linewidth - \l_enumext_minipage_right_v_dim - \l_enumext_minipage_hsep_v_dim
}
```

The boolean variable \l__enumext_minipage_active_v_bool will be activated and the integer variable \g__enumext_minipage_stat_int used by the \miniright command will be incremented, then the function __enumext_keyans_mini_addvspace: is called and the __enumext_mini_env* environment on left side will be initialized followed by the vertical spacing \l__enumext_minipage_left_skip. Here we use the plain TEX macro \nointerlineskip to prevent baseline "glue" being added between the next pair of boxes in a vertical list.

After these actions, the $_$ enumext_keyans_multicols_start: function is called to handle the multicols environment.

```
3089 \__enumext_keyans_multicols_start:
3090 }
```

(End of definition for __enumext_before_list_v:.)

__enumext_keyans_multicols_start:

The function __enumext_keyans_multicols_start: will start the multicols environment according to the value passed by the columns key.

Set the default value for \columnsep when columns-sep key is Opt.

Then we will set the value of \multicolsep and \columnseprule equal to zero (we do not want a vertical rule in this environment).

```
\skip_zero:N \multicolsep
\dim_zero:N \columnseprule
```

We will calculate the *vertical spacing* settings for the multicols environment using the function __enumext_keyans_multi_addvspace: and apply our "*vertical adjust spacing*", then start the multicols environment.

 $(\textit{End of definition for } \verb|_-enumext_keyans_multicols_start:.)$

__enumext_keyans_multicols_stop:

The function __enumext_keyans_multicols_stop: will stop the multicols environment. If the boolean variable \l__enumext_minipage_active_v_bool is false (not nested in __enumext_mini_env*) we will apply our vertical "adjust" spacing.

```
316 \cs_new_protected:Nn \__enumext_keyans_multicols_stop:
317 {
318 \int_compare:nNnT { \l__enumext_columns_v_int } > { 1 }
319 {
310 \end{multicols}
312 \bool_if:NF \l__enumext_minipage_active_v_bool
312 {
312 \par\addvspace{ \l__enumext_multicols_below_v_skip }
312 }
312 }
312 }
312 }
312 }
312 }
312 }
312 }
```

(End of definition for __enumext_keyans_multicols_stop:.)

__enumext_after_list_v:

The function __enumext_after_list_v: will will check the state of the boolean variable \l__enumext_minipage_active_v_bool, if it is "true" a small test will be executed to check if we have omitted the use of \miniright (the __enumext_mini_env* environment has not been closed), then close __enumext_mini_env* and add the vertical adjustment space \l__enumext_minipage_after_skip, otherwise we will close the multicols environment.

Finally we will apply the $\{\langle code \rangle\}$ handled by the after key together with the *vertical space* handled by the below key if they are present.

(End of definition for $_$ enumext_after_list_v:.)

11.33 The environment keyanspic and \anspic

The keyanspic environment is a list-based environment that uses the same configuration for "spacing" and $\langle label \rangle$ as the keyans environment, but it does not use \item.

The contents are passed to the environment by means of the \anspic command and are placed inside minipage environments, with the $\langle label \rangle$ underneath, adjusting widths according to the options passed to the environment.

Again it is necessary to "adjust" the spacing, both vertical and horizontal, to obtain an output like the one shown in the figure 12.

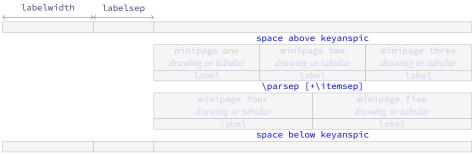


Figure 12: Representation of the keyanspic spacing in enumext.

This implementation is adapted from the answer given by Enrico Gregorio in How to process the body of an environment and divide it by a \macro?.

11.33.1 The command \anspic

\anspic

The \anspic command take three arguments, the starred (*) versions \anspic* and \anspic* [$\langle content \rangle$] store the current $\langle label \rangle$ next to the [$\langle content \rangle$] if it is present in the $\langle sequence \rangle$ and $\langle prop \ list \rangle$ defined by save-ans key. This command is used as a replacement for \item in the keyanspic environment.

```
_{3145} \NewDocumentCommand \anspic { s o +m }
```

We check that the command is active in the keyanspic environment only if the save-ans key is present, otherwise we return an error.

The three arguments are handled by the function __enumext_keyans_anspic_code:nnn and stored in the sequence \l__enumext_keyans_pic_body_seq which is processed by the keyanspic environment.

(End of definition for \anspic. This function is documented on page 13.)

__enumext_keyans_anspic_code:nnn

The function $\ensuremath{\mbox{\mbox{-}enumext_keyans_anspic_code:nnn}}$ will be in charge of handling the "counter" and $\langle label \rangle$, which will have the same configuration as the keyans environment.

```
\cs_new_protected:Nn \__enumext_keyans_anspic_code:nnn
3165
       \stepcounter { enumXvi }
       #3 \\
       \bool_if:nT { #1 }
         {
             _enumext_keyans_addto_prop:n { #2 }
           \__enumext_keyans_store_ref:
           \__enumext_keyans_addto_seq:n { #2 }
           \int_gincr:N \g__enumext_check_starred_cmd_int
           \bool_lazy_or:nnT
             { \bool_if_p:N \l__enumext_show_answer_bool }
               \bool_if_p:N \l__enumext_show_position_bool }
               \tl_set_eq:NN \l__enumext_label_v_tl \l__enumext_label_vi_tl
               \ enumext keyans show left:n { #2 }
               \tl_set_eq:NN \l__enumext_label_vi_tl \l__enumext_label_v_tl
2181
3182
       \tl_use:N \l__enumext_label_font_style_v_tl
3183
         _enumext_wrapper_label_v:n {    \l__enumext_label_vi_tl } \__enumext_keyans_show_item_opt:
3184
3185
```

 $(\textit{End of definition for } \verb|_-enumext_keyans_anspic_code:nnn.)$

11.33.2 The environment keyanspic

keyanspic Now we define the environment keyanspic based on list. The optional argument [\(\lambda number above, number below \rangle \)] will determine the number of minipage environments that will be above and below separated by \parsep+\itemsep within it.

```
\__enumext_keyans_pic_arg_two:
```

We apply the "adjusted" vertical spacing above the environment

```
3194     \vspace { \l__enumext_keyans_pic_above_skip }
3195   }
```

If the optional argument is not present, the number of times the \anspic command appears will be counted from \l__enumext_keyans_pic_body_seq and placed in minipage environments on a single line. Finally we check if \anspic* has been used, set the counter to zero and apply our "adjusted" vertical space below the environment.

(End of definition for keyanspic. This function is documented on page 12.)

__enumext_keyans_pic_safe_exec:

The function __enumext_keyans_pic_safe_exec: check nested and level position inside the enumext environment.

(End of definition for $\ensuremath{\setminus}$ _enumext_keyans_pic_safe_exec:.)

__enumext_keyans_pic_skip_abs:N

The function __enumext_keyans_pic_skip_abs: N will return a positive value \parsep.

 $(\mathit{End}\ of\ definition\ for\ \verb|_enumext_keyans_pic_skip_abs:N.)$

__enumext_keyans_pic_arg_two:

The function __enumext_keyans_pic_arg_two: will be used in the second argument of the __enumext_start_list:nn function that defines the keyanspic environment, it will handle the setting of spaces.

```
3222 \cs_new_protected:Nn \__enumext_keyans_pic_arg_two:
```

The first thing to do is to set the boolean variable \l_enumext_leftmargin_tmp_v_bool handled by the list-indent key to false, then we copy the definition of the second list argument from the keyans environment.

```
\bool_set_false:N \l__enumext_leftmargin_tmp_v_bool \__enumext_list_arg_two_v:
```

We will add the value of \itemsep to \parsep which we will use as vertical spacing between the above and below minipage environments. and adjust the value of \leftmargin, the label and counter are handled directly by the \anspic command. Then we make equal to zero \labelwidth, \labelsep, \partopsep and \itemsep so that the horizontal and vertical spacing is not affected.

```
\lambda \skip_add:\nn \parsep \ \itemsep \}
\dim_add:\nn \leftmargin \ \-\lambda - \lambda \lambda \leftmargin \ \dim_zero:\n\ \lambda \lambda \limbda \mathred \math
```

We set the value of \l_enumext_keyans_pic_above_skip which we will use to apply our "adjust" space above keyanspic, finally we call _enumext_item_std:w followed by \scan_stop: to prevent the error message returned by LTPX when not using the \item command.

```
\__enumext_keyans_pic_skip_abs:N \parsep
\skip_set:Nn \l__enumext_keyans_pic_above_skip

{

\box_dp:N \strutbox
+ \l__enumext_topsep_v_skip
- \parsep
}

\__enumext_item_std:w \scan_stop:

}
```

(End of definition for $\ensuremath{\setminus}$ _enumext_keyans_pic_arg_two:.)

__enumext_keyans_pic_do:n
__enumext_keyans_pic_do:e

The optional argument is split by comma and is handled directly by the function __enumext_keyans_-pic_do:n and passed to the function __enumext_keyans_pic_row:n.

```
3242 \cs_new_protected:Nn \__enumext_keyans_pic_do:n
3243 {
3244 \clist_map_function:nN { #1 } \__enumext_keyans_pic_row:n
3245 }
3246 \cs_generate_variant:Nn \__enumext_keyans_pic_do:n { e }

(End of definition for \__enumext_keyans_pic_do:n.)
```

__enumext_keyans_pic_row:n

The function __enumext_keyans_pic_row:n will set the widths for the minipage environments and place the content $\langle stored \rangle$ by \anspic* in the \l__enumext_keyans_pic_body_seq sequence inside them.

(End of definition for __enumext_keyans_pic_row:n.)

11.34 The environment enumext*

Generating horizontal list environments is NOT as simple as standard LaTeX list environments. The fundamental part of the code is adapted from the shortlst package to a more modern version using expl3. It is not possible to redefine \item and \makelabel as in the non starred versions (at least I have not achieved it) and as we will make it behave differently, we have no other option than to define a cascade of functions.

To achieve the horizontal list environment we will capture the \item command and the content of this in an plain \lambdarbox box using \makebox for the \lambdabel and a minipage environment for the content passed to \item, we will also add the optional argument ($\langle number \rangle$) to \item to be able to join columns horizontally, in simple terms, we want \item to behave in the same way as in the enumext environment but adding an optional first argument ($\langle number \rangle$).

11.34.1 Functions for item box width

__enumext_starred_columns_set_vii:

We set the default value for the width of the box containing the content of the items and create \itemwidth in a public form.

©2024 by Pablo González L

91/125

```
/ \l_enumext_columns_vii_int

/ \l_enumext_tmpa_vii_int { \l_enumext_columns_vii_int - \c_one_int }

/ \int_set:Nn \l_enumext_item_width_vii_dim

/ \l_enumext_columns_sep_vii_dim * \l_enumext_tmpa_vii_int )

/ \l_enumext_columns_vii_int - \l_enumext_labelwidth_vii_dim

- \l_enumext_columns_vii_int - \l_enumext_labelwidth_vii_dim

- \l_enumext_labelsep_vii_dim

- \l_enumext_labelsep_vii_dim

/ \dim_zero_new:N \itemwidth

/ \lefticup \leftilde \left
```

__enumext_starred_joined_item_vii:n

The function $_$ _enumext_starred_joined_item_vii:n will set the *width* of the box in which the content passed to $\ideticontent(\normalfont{number})$ will be stored together with the value of \ideticontent{number} .

```
3282 \cs_new_protected:Npn \__enumext_starred_joined_item_vii:n #1
       \int_set:Nn \l__enumext_joined_item_vii_int {#1}
3284
       \int_compare:nNnT { \l__enumext_joined_item_vii_int } > { \l__enumext_columns_vii_int }
3285
         {
3286
           \msg_warning:nnee { enumext } { item-joined }
3287
             { \int_use:N \l__enumext_joined_item_vii_int }
3288
             { \int_use:N \l__enumext_columns_vii_int }
           \int_set:Nn \l__enumext_joined_item_vii_int
                \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + \c_one_int
         }
       \int compare:nNnT
         { \l__enumext_joined_item_vii_int }
3296
3297
         { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + \c_one_int }
3298
         {
3299
            \msg_warning:nnee { enumext } { item-joined-columns }
3300
             { \int_use:N \l__enumext_joined_item_vii_int }
3301
             {
                \int_eval:n
                  { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + \c_one_int }
             }
           \int_set:Nn \l__enumext_joined_item_vii_int
3306
             {
3307
                \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + \c_one_int
3308
3309
         }
3310
```

Only need if #1 » 1 (default are set before).

```
\int_compare:nNnTF { \l__enumext_joined_item_vii_int } > { \c_one_int }
         {
3312
           \int_set_eq:NN \l__enumext_joined_item_aux_vii_int \l__enumext_joined_item_vii_int
           \int_decr:N \l__enumext_joined_item_aux_vii_int
           \int_add:Nn \l__enumext_item_column_pos_vii_int { \l__enumext_joined_item_aux_vii_int }
           \int_gadd:Nn \g__enumext_item_count_all_vii_int { \l__enumext_joined_item_aux_vii_int }
           \dim_set:Nn \l__enumext_joined_width_vii_dim
             {
               \l__enumext_item_width_vii_dim * \l__enumext_joined_item_vii_int
               + ( \l__enumext_labelwidth_vii_dim + \l__enumext_labelsep_vii_dim
                  + \l__enumext_columns_sep_vii_dim
                 )*\l__enumext_joined_item_aux_vii_int
           \dim_set_eq:NN \itemwidth \l__enumext_joined_width_vii_dim
         }
         {
           \dim_set_eq:NN \l__enumext_joined_width_vii_dim \l__enumext_item_width_vii_dim
           \dim_set_eq:NN \itemwidth \l__enumext_item_width_vii_dim
3328
3330
```

 $(\textit{End of definition for } \verb|_-enumext_starred_joined_item_vii:n.)$

__enumext_start_mini_vii:

The implementation of the mini-env key support is almost identical to the one used in the enumext and keyans environments, the difference is that the __enumext_mini_env* environment on the "right side" is executed "after" closing the environment, so it is necessary to make a global copy of the variable \l_enumext_minipage_right_vii_dim in the variable \g_enumext_minipage_right_vii_dim.

```
3331 \cs_new_protected:Nn \__enumext_start_mini_vii:
     {
3332
       \dim_compare:nNnT { \l__enumext_minipage_right_vii_dim } > { \c_zero_dim }
           \dim_set:Nn \l__enumext_minipage_left_vii_dim
             {
               \linewidth
               - \l__enumext_minipage_right_vii_dim
                - \l__enumext_minipage_hsep_vii_dim
           \bool_set_true:N \l__enumext_minipage_active_vii_bool
3341
           \dim_gset_eq:NN
3342
             \g__enumext_minipage_right_vii_dim
3343
             \l__enumext_minipage_right_vii_dim
           \__enumext_mini_addvspace_vii:
           \nointerlineskip\noindent
           \begin{__enumext_mini_env*}{ \l__enumext_minipage_left_vii_dim }
         }
      }
3349
```

(End of definition for __enumext_start_mini_vii:.)

__enumext_stop_mini_vii:

The function __enumext_stop_mini_vii: closes the __enumext_mini_env* environment on the left side, applies \hfill and sets the value of the variable \g__enumext_minipage_active_vii_bool to true which will be used in the function __enumext_after_star_env:nn to execute the __enumext_mini_env* on the "right side".

Finally we execute code passed to the miniright key stored in the variable \g__enumext_miniright_-code_vii_tl in the __enumext_mini_env* environment on the "right side".

```
\__enumext_after_env:nn {enumext*}
       \bool_if:NT \g__enumext_minipage_active_vii_bool
           \begin{__enumext_mini_env*}{ \g__enumext_minipage_right_vii_dim }
             \par\addvspace { \g__enumext_minipage_right_skip }
             \bool_if:NF \g__enumext_minipage_center_vii_bool
                 \centering
             \tl_use:N \g__enumext_miniright_code_vii_tl % the code
           \end{__enumext_mini_env*}
           \par\addvspace{ \g_enumext_minipage_after_skip }
         }
       \bool_gset_false:N \g__enumext_minipage_active_vii_bool
       \bool_gset_true:N \g__enumext_minipage_center_vii_bool
3374
       \tl_gclear:N \g__enumext_miniright_code_vii_tl
       \dim_gzero:N \g__enumext_minipage_right_vii_dim
       \bool_gset_false:N \g__enumext_starred_bool
3377
3378
```

(End of definition for __enumext_stop_mini_vii:.)

First we will generate the environment and we will give a temporary definition to __enumext_stop_-item_tmp_vii: equal to \noindent and next to \item equal to __enumext_start_item_tmp_vii: which we will redefine later.

```
3379 \NewDocumentEnvironment{enumext*}{ o }
3380 {

©2024 by Pablo González L
```

```
\__enumext_safe_exec_vii:
        \__enumext_parse_keys_vii:n {#1}
        \__enumext_before_list_vii:
3383
        \__enumext_start_store_level_vii:
3384
        \__enumext_start_list:nn { }
3385
          {
3386
            \__enumext_list_arg_two_vii:
3387
            \__enumext_before_keys_exec_vii:
3388
          }
3389
          \__enumext_starred_columns_set_vii:
          \item[] \scan_stop:
          \cs_set_eq:NN \__enumext_stop_item_tmp_vii: \noindent
          \cs_set_eq:NN \item \__enumext_start_item_tmp_vii:
     }
3394
3395
        \__enumext_stop_item_tmp_vii:
3396
        \__enumext_remove_extra_parsep_vii:
3397
        \__enumext_stop_list:
3398
        \__enumext_stop_store_level_vii:
3399
        \__enumext_after_list_vii:
(End of definition for enumext*. This function is documented on page 4.)
```

__enumext_safe_exec_vii: First check the maximum nesting level for the enumext* environment then set the vars \l__enumext_starred_bool and \g__enumext_starred_bool.

```
3402 \cs_new_protected:Nn \__enumext_safe_exec_vii:
    {
3403
       \__enumext_is_not_nested:
3404
       \int_incr:N \l__enumext_level_h_int
       \int_compare:nNnT { \l__enumext_level_h_int } > { 1 }
         {
           \msg_error:nn { enumext } { nested }
       \bool_set_true:N \l__enumext_starred_bool
3410
       \__enumext_is_on_first_level:
3411
3412
```

__enumext_parse_keys_vii:n

Parse $[\langle key = val \rangle]$ for enumext*. If the variable \l__enumext_store_active_bool is true it will call the functions __enumext_parse_serie:n and __enumext_store_active_keys_vii:n and reprocess the keys to pass them to the storage *(sequence)*.

```
3413 \cs_new_protected:Npn \__enumext_parse_keys_vii:n #1
       \tl_if_novalue:nF {#1}
3415
         {
3416
            \str_clear:N \l__enumext_series_str
3417
            \keys_set:nn { enumext / enumext* } {#1}
3418
            \__enumext_parse_series:n {#1}
3419
            \__enumext_store_active_keys_vii:n {#1}
3420
         }
3421
     }
```

(End of definition for __enumext_parse_keys_vii:n.)

__enumext_before_list_vii:

The function __enumext_before_list_vii: will add the vertical spacing on the environment if the above key is active next to the $\{\langle code \rangle\}$ defined by the before* key if it is active, the call the function __enumext_start_mini_vii: handle by mini-env.

```
3423 \cs_new_protected:Nn \__enumext_before_list_vii:
3424
       \__enumext_vspace_above_vii:
3425
       \__enumext_check_ans_active:
3426
       \__enumext_before_args_exec_vii:
3427
       \__enumext_start_mini_vii:
3428
     }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_before_list_vii:.)$

__enumext_after_list_vii:

The function __enumext_after_list: first call the function __enumext_stop_mini_vii:, then apply the $\{\langle code \rangle\}$ handled by the after key together with the $vertical\ space$ handled by the below key if they are present. Finally set false the vars \g__enumext_starred_bool and \l__enumext_starred_bool, save the $current\ value$ of the counter in \g__enumext_resume_vii_int for the resume key. If the save-ans key is active, it will create the integer variable for the resume key, we only have to assign it the value of the current counter.

```
3430 \cs_new_protected:Nn \__enumext_after_list_vii:
3431 {
3432 \__enumext_stop_mini_vii:
3433 \__enumext_after_stop_list_vii:
3434 \__enumext_check_ans_key_hook:
3435 \__enumext_vspace_below_vii:
3436 \bool_set_false:N \l__enumext_starred_bool
3437 \__enumext_resume_save_counter:
3438 }
```

(End of definition for $_$ enumext_after_list_vii:.)

__enumext_start_store_level_vii:
__enumext_stop_store_level_vii:

The __enumext_start_store_level_vii: and __enumext_stop_store_level_vii: functions activate the level saving mechanism for storage in $\langle sequence \rangle$ of the \anskey command if enumext* are nested in enumext.

```
3439 \cs_new_protected:Nn \__enumext_start_store_level_vii:
     {
3440
       \bool_if:NT \l__enumext_store_active_bool
3441
3442
            \int_compare:nNnT { \l__enumext_level_int } > { \c_zero_int }
                \__enumext_store_level_open_vii:
         }
   \cs_new_protected:Nn \__enumext_stop_store_level_vii:
3449
3450
       \bool_if:NT \l__enumext_store_active_bool
3451
3452
            \int_compare:nNnT { \l__enumext_level_int } > { \c_zero_int }
3453
                \__enumext_store_level_close_vii:
         }
3458
```

 $(\textit{End of definition for } \\ _\texttt{enumext_start_start_store_level_vii:} \ \textit{and } \\ \\ _\texttt{enumext_stop_store_level_vii:})$

11.34.2 The command \item in enumext*

__enumext_start_item_tmp_vii:

First we will call the function __enumext_stop_item_tmp_vii: that we will redefine later, we will increment the value of \l_enumext_item_column_pos_vii_int that will count the item's by rows and the value of \g_enumext_item_count_all_vii_int that will count the total of item's in the environment. After that we will call the function __enumext_item_peek_args_vii: that will handle the arguments passed to \item.

(End of definition for __enumext_start_item_tmp_vii:.)

__enumext_item_peek_args_vii:

The function __enumext_item_peek_args_vii: will handle the \item($\langle number \rangle$). Look for the argument "(", if it is present we will call the function __enumext_joined_item_vii:w ($\langle number \rangle$), which is in charge of joining the item's in the same row, in case they are not present we will set the default value (1).

__enumext_joined_item_vii:w

The function __enumext_joined_item_vii:w will first call the function __enumext_starred_-joined_item_vii:n in charge of setting the width of the box that will store the content passed to \item. Then we will look for the argument "*", if it is present we will call the function __enumext_starred_-item_vii:w otherwise we will call the function __enumext_standar_item_vii:w.

```
3472 \cs_new_protected:Npn \__enumext_joined_item_vii:w (#1)
3473 {
3474 \__enumext_starred_joined_item_vii:n {#1}
3475 \peek_meaning_remove:NTF *
3476 {\__enumext_starred_item_vii:w }
3477 {\__enumext_standar_item_vii:w }
3478 }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_joined_item_vii:w.|)$

__enumext_standar_item_vii:w

The function __enumext_standar_item_vii:w will first look for the argument "[", if present it will set the state of the variable \l__enumext_wrap_label_opt_vii_bool equal to the state of the variable \l__enumext_wrap_label_opt_vii_bool handled by the key wrap-label* and finally execute the non-enumerated version \item[\langle custom \rangle] by means of the function __enumext_start_item_vii:w, otherwise we will set the value of the variable \l__enumext_wrap_label_vii_bool handled by the wrap-label key to true and set the switch \if@noitemarg to true to execute the enumerated version of \item by means of the function __enumext_start_item_vii:w [\l__enumext_label_vii_tl].

```
3479 \cs_new_protected:Npn \__enumext_standar_item_vii:w
3480
     {
       \bool_set_false:N \l__enumext_item_starred_vii_bool
3481
         \peek_meaning:NTF [
              \bool_set_eq:NN
                \l__enumext_wrap_label_vii_bool
                \l__enumext_wrap_label_opt_vii_bool
3486
              \ enumext start item vii:w
3487
           }
3488
3489
              \bool_set_true:N \l__enumext_wrap_label_vii_bool
              \legacy_if_set_true:n { @noitemarg }
3491
              \__enumext_start_item_vii:w [ \l__enumext_label_vii_tl ]
           }
3493
     }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|__enumext_standar_item_vii:w.)$

__enumext_starred_item_vii:w
__enumext_starred_item_vii_aux_i:w
__enumext_starred_item_vii_aux_ii:w
__enumext_starred_item_vii_aux_iii:w

The function __enumext_starred_item_vii:w together with the specified auxiliary functions aux_i:w, aux_ii:w, and aux_iii:w execute \item*, \item*[$\langle symbol \rangle$] and \item*[$\langle symbol \rangle$] [$\langle offset \rangle$].

```
3495 \cs_new_protected:Npn \__enumext_starred_item_vii:w
       \bool_set_true:N \l__enumext_item_starred_vii_bool
3497
       \bool_set_true:N \l__enumext_wrap_label_vii_bool
       \peek_meaning:NTF [
         { \__enumext_starred_item_vii_aux_i:w }
         { \__enumext_starred_item_vii_aux_ii:w }
3501
3502
3503 \cs_new_protected:Npn \__enumext_starred_item_vii_aux_i:w [#1]
3504
       \tl_gset:Nn \g__enumext_item_symbol_aux_vii_tl {#1}
3505
       \__enumext_starred_item_vii_aux_ii:w
3506
     }
3507
3508 \cs_new_protected:Npn \__enumext_starred_item_vii_aux_ii:w
       \peek_meaning:NTF [
3510
         { \__enumext_starred_item_vii_aux_iii:w }
           \dim set ea:NN
             \l__enumext_item_symbol_sep_vii_dim
              \l__enumext_labelsep_vii_dim
           \legacy_if_set_true:n { @noitemarg }
             _enumext_start_item_vii:w [ \l__enumext_label_vii_tl ]
3520 \cs_new_protected:Npn \__enumext_starred_item_vii_aux_iii:w [#1]
©2024 by Pablo González L
```

96 / 125

(End of definition for __enumext_starred_item_vii:w and others.)

11.34.3 Real definition of \item in enumext*

__enumext_start_item_vii:w

The functions __enumext_start_item_vii: w and __enumext_stop_item_vii: executing the true definition of \item inside the enumext* environment.

The first thing we will do is set the value of __enumext_stop_item_tmp_vii: equal to the value of __enumext_stop_item_vii: which we will define later and add the hyperref compatible enumXvii counter, after that we will start capturing the item content in a box. Here need setting the \if@hyper@item switch to "true" for hyperref compatible. The explanation for this is given by the master Heiko Oberdiek on \refstepcounter{enumi} twice (or more) creates destination with the same identifier.

```
3526 \cs_new_protected_nopar:Npn \__enumext_start_item_vii:w [#1]
    {
       \cs_set_eq:NN \__enumext_stop_item_tmp_vii: \__enumext_stop_item_vii:
       \legacy_if:nT { @noitemarg }
         {
           \legacy_if_set_false:n { @noitemarg }
           \legacy_if:nT { @nmbrlist }
             {
               \bool_if:NT \l__enumext_hyperref_bool
                 {
                   \legacy_if_set_true:n { @hyper@item }
                 }
               \refstepcounter{enumXvii}
3538
               \bool_if:NT \l__enumext_check_answers_bool
3540
                   \int_gincr:N \g__enumext_item_number_int
354
                 }
             }
```

Here we start capturing \item and its contents into a group using the plain form of the \lambda rovironment. If the state of the variable \l__enumext_footnotes_key_bool is false, we will redefine the command \footnote, followed by printing the $\langle symbol \rangle$ defined for \item* if it is present and open a new group inside which we execute font key next to \item and the keys wrap-label, wrap-label*, align, close the group and execute the key labelsep and then the key first. Finally we open the minipage environment and execute the listparindent key which will be equal to \parindent, the parsep key which will be equal to \parindent key and the itemindent key.

```
\group_begin:
         \lrbox{ \l__enumext_item_text_vii_box }
3546
           \bool_if:NF \l__enumext_footnotes_key_bool
3547
             {
               \__enumext_renew_footnote:
             }
           \bool_if:NT \l__enumext_item_starred_vii_bool
             {
               \tl_if_blank:VT \g__enumext_item_symbol_aux_vii_tl
                 {
                   \tl_gset_eq:NN
                     \g__enumext_item_symbol_aux_vii_tl \l__enumext_item_symbol_vii_tl
               \mode_leave_vertical:
               \skip_horizontal:n { -\l__enumext_item_symbol_sep_vii_dim }
               \makebox[ Opt ][ r ]{ \g__enumext_item_symbol_aux_vii_tl }
               \skip_horizontal:N \l__enumext_item_symbol_sep_vii_dim
               \tl_gclear:N \g__enumext_item_symbol_aux_vii_tl
             }
           \group_begin:
             \tl_use:N \l__enumext_label_font_style_vii_tl
             \bool_if:NTF \l__enumext_wrap_label_vii_bool
               {
                 \makebox[ \l__enumext_labelwidth_vii_dim ][ \l__enumext_align_label_vii_str ]
                   { \__enumext_wrapper_label_vii:n {#1} }
               }
               {
```

__enumext_stop_item_vii:

The function __enumext_stop_item_vii: shall terminate with the capture of \item and its \(\chiontents \). Close the environments minipage, lrbox and the group. Then we only have to set the width of the box and print it next to \footnote, and add the horizontal and vertical separation between the boxes.

```
3582 \cs_new_protected_nopar:Nn \__enumext_stop_item_vii:
     {
3583
            \__enumext_endminipage:
3584
         \endlrbox
3585
       \group end:
3586
       \box_set_wd:Nn \l__enumext_item_text_vii_box
3587
            \l__enumext_joined_width_vii_dim
            + \l__enumext_labelwidth_vii_dim
            + \l__enumext_labelsep_vii_dim
         }
       \int_set:Nn \hbadness { 10000 }
3593
       \box_use:N \l__enumext_item_text_vii_box
3594
       \bool_if:NF \l__enumext_footnotes_key_bool
3595
         {
3596
            \__enumext_print_footnote:
3597
         }
3598
       \int_compare:nNnTF { \l__enumext_item_column_pos_vii_int } = { \l__enumext_columns_vii_int }
3599
            \par\noindent
            \int_zero:N \l__enumext_item_column_pos_vii_int
3602
         { \hspace{ \l__enumext_columns_sep_vii_dim } }
3604
     }
3605
```

__enumext_remove_extra_parsep_vii:

Finally we will remove the vertical space equal to \parsep when the total number of items is divisible by the number of items in the last row of the environment.

```
3606 \cs_new_protected:Nn \__enumext_remove_extra_parsep_vii:
     {
3607
       \int_compare:nNnT
3608
         {
3609
            \int_mod:nn { \g_enumext_item_count_all_vii_int } { \l_enumext_columns_vii_int }
         }
         =
          { \c_zero_int }
         {
3614
3615
            \vspace{ -\l__enumext_itemsep_vii_skip }
3616
            \int_gzero:N \g__enumext_item_count_all_vii_int
3617
3618
3619
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_remove_extra_parsep_vii:.)$

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_stop_item_vii:.)$

As we don't want our check to be executed check-ans by levels but on the complete list, we will take it out of the enumext* environment using the "hook" function __enumext_after_env:nn.

```
3620 \__enumext_after_env:nn {enumext*} { \__enumext_execute_after_env: }
```

11.35 The environment keyans*

11.35.1 Functions for item box width

__enumext_starred_columns_set_viii:

We set the default value for the width of the box containing the content of the items and create \itemwidth in a public form.

```
3621 \cs_new_protected:Nn \__enumext_starred_columns_set_viii:
       \dim_compare:nNnT { \l__enumext_columns_sep_viii_dim } = { \c_zero_dim }
3623
           \dim_set:Nn \l__enumext_columns_sep_viii_dim
3625
             {
3626
               ( \l__enumext_labelwidth_viii_dim + \l__enumext_labelsep_viii_dim )
3627
                / \l__enumext_columns_viii_int
3628
3630
       \int_set:Nn \l__enumext_tmpa_viii_int { \l__enumext_columns_viii_int - \c_one_int }
3631
       \dim_set:Nn \l__enumext_item_width_viii_dim
           ( \label{linewidth} - \l_enumext\_columns\_sep\_viii\_dim * \l_enumext\_tmpa\_viii\_int )
           / \l__enumext_columns_viii_int - \l__enumext_labelwidth_viii_dim
            - \l__enumext_labelsep_viii_dim
3636
3637
       \dim_zero_new:N \itemwidth
3638
3639
```

(End of definition for __enumext_starred_columns_set_viii:.)

__enumext_starred_joined_item_viii:n

The function __enumext_starred_joined_item_viii:n will set the *width* of the box in which the content passed to \item(\(\lamber\)\)) will be stored together with the value of \item\(\lamber\)

```
content passed to \forall item(\langle number \rangle) will be stored together with the value of \forall itemwidth.
3640 \cs_new_protected:Npn \__enumext_starred_joined_item_viii:n #1
3641
       \int_set:Nn \l__enumext_joined_item_viii_int {#1}
3642
       \int_compare:nNnT { \l__enumext_joined_item_viii_int } > { \l__enumext_columns_viii_int }
3643
3644
            \msg_warning:nnee { enumext } { item-joined }
             { \int_use:N \l__enumext_joined_item_viii_int }
             { \int_use:N \l__enumext_columns_viii_int }
           \int_set:Nn \l__enumext_joined_item_viii_int
                \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + \c_one_int
3651
         }
2652
       \int_compare:nNnT
3653
         { \l__enumext_joined_item_viii_int }
3654
3655
         { \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + \c_one_int }
3656
         {
3657
           \msg_warning:nnee { enumext } { item-joined-columns }
             { \int_use:N \l__enumext_joined_item_viii_int }
             {
                \int eval:n
                  { \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + \c_one_int
3663
           \int_set:Nn \l__enumext_joined_item_viii_int
2664
             {
                \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + \c_one_int
       \int_compare:nNnTF { \l__enumext_joined_item_viii_int } > { \c_one_int }
         {
           \int_set_eq:NN \l__enumext_joined_item_aux_viii_int \l__enumext_joined_item_viii_int
3671
           \int_decr:N \l__enumext_joined_item_aux_viii_int
2672
           \int_add:Nn \l__enumext_item_column_pos_viii_int { \l__enumext_joined_item_aux_viii_int }
3673
            \int_gadd:Nn \g__enumext_item_count_all_viii_int { \l__enumext_joined_item_aux_viii_int }
3674
           \dim_set:Nn \l__enumext_joined_width_viii_dim
3675
3676
                \l__enumext_item_width_viii_dim * \l__enumext_joined_item_viii_int
3677
                + ( \l__enumext_labelwidth_viii_dim + \l__enumext_labelsep_viii_dim
                   + \l__enumext_columns_sep_viii_dim
                  )*\l__enumext_joined_item_aux_viii_int
```

__enumext_start_mini_viii:
__enumext_stop_mini_viii:

```
\dim_set_eq:NN \itemwidth \l__enumext_joined_width_viii_dim
         }
3684
         {
            \dim_set_eq:NN \l__enumext_joined_width_viii_dim \l__enumext_item_width_viii_dim
3685
            \dim_set_eq:NN \itemwidth \l__enumext_item_width_viii_dim
3686
3687
3688
(End of definition for \__enumext_starred_joined_item_viii:n.)
The implementation of the mini-env key is identical to the one used in the enumext* environment.
3689 \cs_new_protected:Nn \__enumext_start_mini_viii:
       \dim_compare:nNnT { \l__enumext_minipage_right_viii_dim } > { \c_zero_dim }
3691
         {
3692
            \dim_set:Nn \l__enumext_minipage_left_viii_dim
3693
              {
3694
                \linewidth
3695
                - \l__enumext_minipage_right_viii_dim
                - \l__enumext_minipage_hsep_viii_dim
            \bool_set_true:N \l__enumext_minipage_active_viii_bool
            \dim_gset_eq:NN
              \g__enumext_minipage_right_viii_dim
              \l__enumext_minipage_right_viii_dim
            \__enumext_mini_addvspace_viii:
            \nointerlineskip\noindent
            \begin{__enumext_mini_env*}{ \l__enumext_minipage_left_viii_dim }
3706
      }
   \cs_new_protected:Nn \__enumext_stop_mini_viii:
       \bool_if:NT \l__enumext_minipage_active_viii_bool
         {
            \end{__enumext_mini_env*}
3712
            \hfill
            \bool_gset_true:N \g__enumext_minipage_active_viii_bool
3716
    \__enumext_after_env:nn {keyans*}
3718
       \bool_if:NT \g__enumext_minipage_active_viii_bool
3719
            \begin{__enumext_mini_env*}{ \g__enumext_minipage_right_viii_dim }
              \par\addvspace { \g__enumext_minipage_right_skip }
              \bool_if:NF \g__enumext_minipage_center_viii_bool
                {
3724
                  \centering
              \tl_use:N \g__enumext_miniright_code_viii_tl % the code
            \end{__enumext_mini_env*}
3728
            \par\addvspace{ \g__enumext_minipage_after_skip }
3729
3730
       \bool_gset_false:N \g__enumext_minipage_active_viii_bool
       \bool_gset_true:N \g__enumext_minipage_center_viii_bool
       \tl_gclear:N \g__enumext_miniright_code_viii_tl
       \dim_gzero:N \g__enumext_minipage_right_viii_dim
3734
(End of definition for \__enumext_start_mini_viii: and \__enumext_stop_mini_viii:.)
First we will generate the environment and we will give a temporary definition to \__enumext_stop_-
item_tmp_viii: equal to \noindent and next to \item equal to \__enumext_start_item_tmp_-
viii: which we will redefine later.
3736 \NewDocumentEnvironment{keyans*}{ o }
       \__enumext_safe_exec_viii:
3738
       \__enumext_parse_keys_viii:n {#1}
3739
       \__enumext_before_list_viii:
3740
       \__enumext_start_list:nn { }
3741
```

```
_enumext_list_arg_two_viii:
                                                _enumext_before_keys_exec_viii:
                                           }
                                           \__enumext_starred_columns_set_viii:
                                           \item[] \scan_stop:
                                           \cs_set_eq:NN \__enumext_stop_item_tmp_viii: \noindent
                                 3748
                                           \cs_set_eq:NN \item \__enumext_start_item_tmp_viii:
                                         \__enumext_stop_item_tmp_viii:
                                         \__enumext_remove_extra_parsep_viii:
                                         \__enumext_check_starred_cmd:n { item }
                                         \__enumext_stop_list:
                                         \__enumext_after_list_viii:
                                 3756
                                 (End of definition for keyans*. This function is documented on page 11.)
                                 First check the maximum nesting level for the keyans* environment.
  \ enumext safe exec viii:
                                 3758 \cs_new_protected:Nn \__enumext_safe_exec_viii:
                                 3759
                                         \int_incr:N \l__enumext_keyans_level_h_int
                                 3760
                                         \int_compare:nNnT { \l__enumext_keyans_level_h_int } > { 1 }
                                 3761
                                              \msg_error:nn { enumext } { nested }
                                           }
                                         \__enumext_keyans_save_start_line:
                                         % Set false for interfering with enumext nested in keyans* (yes, its possible and crayze)
                                 3766
                                         \bool_set_false:N \l__enumext_store_active_bool
                                 3767
                                         \int_compare:nNnT { \l__enumext_level_int } > { 1 }
                                 3768
                                 3769
                                              \msg_error:nn { enumext } { keyans-wrong-level }
                                           }
                                 3771
                                 (End of definition for \__enumext_safe_exec_viii:.)
                                 Parse [\langle key = val \rangle] for keyans*.
\__enumext_parse_keys_viii:n
                                 3773 \cs_new_protected:Npn \__enumext_parse_keys_viii:n #1
                                         \tl_if_novalue:nF {#1}
                                           {
                                              \keys_set:nn { enumext / keyans* } {#1}
                                 3778
                                 (End of definition for \__enumext_parse_keys_viii:n.)
                                 The function \__enumext_before_list_viii: will add the vertical spacing on the environment if the
\__enumext_before_list_viii:
                                 above key is active next to the \{\langle code \rangle\} defined by the before* key if it is active, the call the function
                                 \__enumext_start_mini_viii: handle by mini-env.
                                 3780 \cs_new_protected:Nn \__enumext_before_list_viii:
                                            _enumext_vspace_above_viii:
                                         \__enumext_before_args_exec_viii:
                                         \__enumext_start_mini_viii:
                                 (\mathit{End}\ of\ definition\ for\ \verb|\_-enumext\_before\_list\_viii:.)
                                 The function \__enumext_after_list: first call the function \__enumext_stop_mini_viii:, then
 \__enumext_after_list_viii:
                                 apply the \{\langle code \rangle\} handled by the after key together with the vertical space handled by the below key if
                                 they are present.
                                 3786 \cs_new_protected:Nn \__enumext_after_list_viii:
                                 3787
                                         \__enumext_stop_mini_viii:
                                         \__enumext_after_stop_list_viii:
                                         \__enumext_vspace_below_viii:
                                 (End of definition for \_enumext_after_list_viii:.)
                                 ©2024 by Pablo González L
```

101/125

11.35.2 The command \item in keyans*

The idea here is to make the \item command behave in the same way as in the keyans environment with the difference of the optional argument $(\langle number \rangle)$ which works in the same way as in the enumext* environment. In simple terms we want to store the $\langle label \rangle$ next to the $\lceil \langle content \rangle \rceil$ if it is present in the $\langle sequence \rangle$ and $\langle prop \ list \rangle$ defined by save-ans key for \item*, \item* $\lceil \langle content \rangle \rceil$, \item($\langle number \rangle$)* and \item($\langle number \rangle$)* $\lceil \langle content \rangle \rceil$ commands.

__enumext_start_item_tmp_viii:

First we will call the function __enumext_stop_item_tmp_viii: that we will redefine later, we will increment the value of \l__enumext_item_column_pos_viii_int that will count the item's by rows and the value of \g__enumext_item_count_all_viii_int that will count the total of item's in the environment. After that we will call the function __enumext_item_peek_args_viii: that will handle the arguments passed to \item.

```
3792 \cs_new_protected_nopar:Nn \__enumext_start_item_tmp_viii:
3793 {
3794 \__enumext_stop_item_tmp_viii:
3795 \int_incr:N \l__enumext_item_column_pos_viii_int
3796 \int_gincr:N \g__enumext_item_count_all_viii_int
3797 \__enumext_item_peek_args_viii:
3798 }
```

 $(\textit{End of definition for } \verb|__enumext_start_item_tmp_viii:.)$

__enumext_item_peek_args_viii:

The function __enumext_item_peek_args_viii: will handle the \item($\langle number \rangle$). Look for the argument "(", if it is present we will call the function __enumext_joined_item_viii:w ($\langle number \rangle$), which is in charge of joining the item's in the same row, in case they are not present we will set the default value (1).

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_item_peek_args_viii:.)$

__enumext_joined_item_viii:w

The function __enumext_joined_item_viii:w will first call the function __enumext_starred_-joined_item_viii:n in charge of setting the *width* of the box that will store the content passed to \item. Then we will look for the argument "*", if it is present we will call the function __enumext_starred_-item_viii:w otherwise we will call the function __enumext_standar_item_viii:w.

```
3805 \cs_new_protected:Npn \__enumext_joined_item_viii:w (#1)
3806 {
3807 \__enumext_starred_joined_item_viii:n {#1}
3808 \peek_meaning_remove:NTF *
3809 { \__enumext_starred_item_viii:w }
3810 { \__enumext_standar_item_viii:w }
3811 }
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_joined_item_viii:w.|)$

\ enumext standar item viii:w

The function __enumext_standar_item_viii:w will first look for the argument "[", if present it will set the state of the variable \l__enumext_wrap_label_opt_viii_bool equal to the state of the variable \l__enumext_wrap_label_opt_viii_bool handled by the key wrap-label* and finally execute the non-enumerated version \item[$\langle custom \rangle$] by means of the function __enumext_start_item_viii:w, otherwise we will set the value of the variable \l_enumext_wrap_label_viii_bool handled by the wrap-label key to true and set the switch \if@noitemarg to true to execute the enumerated version of \item by means of the function __enumext_start_item_viii:w [\l_enumext_label_viii_tl

```
].

3812 \cs_new_protected:Npn \__enumext_standar_item_viii:w

3813 {

3814 \bool_set_false:N \l__enumext_item_starred_viii_bool

3815 \peek_meaning:NTF [

3816 {

3817 \bool_set_eq:NN

3818 \l__enumext_wrap_label_viii_bool

3819 \l__enumext_wrap_label_opt_viii_bool

3820 \__enumext_start_item_viii:w

3821 }

3822 {

bool_set_true:N \l__enumext_wrap_label_viii_bool

3823 \bool_set_true:N \l_enumext_wrap_label_viii_bool
```

©2024 by Pablo González L

102 / 125

(End of definition for $\ensuremath{\backslash}$ _enumext_standar_item_viii:w.)

__enumext_starred_item_viii:w __enumext_starred_item_viii_aux_ii:w __enumext_starred_item_viii_aux_ii:w The function __enumext_starred_item_viii:w together with the specified auxiliary functions aux_i:w and aux_ii:w execute \item* and \item*[$\langle content \rangle$].

The function __enumext_starred_item_viii_aux_i:w will save the optional argument to \item* in \l__enumext_keyans_item_opt_tl and will save this argument along with the spacing set by the key save-sep in variable \l__enumext_store_keyans_label_tl if present, then call the function __enumext_starred_item_viii_aux_ii:w.

```
3836 \cs_new_protected:Npn \__enumext_starred_item_viii_aux_i:w [#1]
    {
3837
      \tl_clear:N \l__enumext_store_keyans_label_tl
3838
      \tl_if_novalue:nF { #1 }
3839
          \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
              \tl_put_right:Ne \l__enumext_store_keyans_label_tl { \l__enumext_store_keyans_item_op
              \tl_put_right:Ne \l__enumext_store_keyans_label_tl { #1 }
          \tl_set:Ne \l__enumext_keyans_item_opt_tl { #1 }
3846
3847
3848
        _enumext_starred_item_viii_aux_ii:w
    }
3849
  \cs_new_protected:Npn \__enumext_starred_item_viii_aux_ii:w
3851
    {
      \legacy_if_set_true:n { @noitemarg }
        _enumext_start_item_viii:w [ \l__enumext_label_viii_tl ]
3853
    }
3854
```

__enumext_starred_item_exec:

starred_item_viii_aux_ii:w.)

The function __enumext_starred_item_exec: will be in charge of storing the current $\langle label \rangle$ for \item* followed by the $[\langle content \rangle]$ for \item* $[\langle content \rangle]$ if present in the $\langle sequence \rangle$ and $\langle prop \ list \rangle$ set by the save-ans key. In this same function the keys show-ans, show-pos and save-ref are implemented.

```
3855 \cs_new_protected:Nn \__enumext_starred_item_exec:
3856
       \tl_put_left:Ne \l__enumext_store_keyans_label_tl { \l__enumext_label_viii_tl }
3857
       \__enumext_store_addto_prop:V \l__enumext_store_keyans_label_tl
3858
       \__enumext_keyans_store_ref:
       \tl_put_left:Ne \l__enumext_store_keyans_label_tl { \item }
       \__enumext_keyans_addto_seq_link:
       \int_gincr:N \g__enumext_check_starred_cmd_int
       \bool_if:NT \l__enumext_show_answer_bool
              _enumext_print_keyans_box:NN \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
3865
3866
       \bool_if:NT \l__enumext_show_position_bool
3867
3868
           \tl_set:Ne \l__enumext_mark_answer_sym_tl
                \group_begin:
                  \exp_not:N \normalfont
                  \exp_not:N \footnotesize [ \int_eval:n
                      \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
                    }
©2024 by Pablo González L
                                                                                                 103 / 125
```

(End of definition for __enumext_starred_item_exec:.)

Real definition of \item in keyans*

__enumext_start_item_viii:w

The implementation at this point is very similar to that of the enumext* environment.

Here we start capturing \item and its contents into a group using the plain form of the lrbox environment.

```
\group_begin:
         \lrbox{ \l__enumext_item_text_viii_box }
           \bool_if:NF \l__enumext_footnotes_key_bool
               \__enumext_renew_footnote:
             }
           \bool_if:NT \l__enumext_item_starred_viii_bool
             {
               \__enumext_starred_item_exec:
             }
           \group_begin:
             \tl_use:N \l__enumext_label_font_style_viii_tl
             \bool_if:NTF \l__enumext_wrap_label_viii_bool
                 \makebox[ \l__enumext_labelwidth_viii_dim ][ \l__enumext_align_label_viii_str ]
                   { \__enumext_wrapper_label_viii:n {#1} }
               }
               {
                 \makebox[ \l__enumext_labelwidth_viii_dim ][ \l__enumext_align_label_viii_str ]{ #1
               }
           \group_end:
           \skip_horizontal:N \l__enumext_labelsep_viii_dim
           \tl_use:N \l__enumext_after_list_args_viii_tl
           \__enumext_minipage:w [ t ]{ \l__enumext_joined_width_viii_dim }
             \skip_set_eq:NN \parindent \l__enumext_listparindent_viii_dim
             \skip_set_eq:NN \parskip \l__enumext_parsep_viii_skip
             \bool_if:NT \l__enumext_item_starred_viii_bool
                 \tl_use:N \l__enumext_fake_item_indent_viii_tl
                 \__enumext_keyans_show_item_opt: \skip_horizontal:n { -\l__enumext_fake_item_indent
3927
               {
                 \tl_use:N \l__enumext_fake_item_indent_viii_tl
3930
               }
3931
```

(End of definition for $\ensuremath{\backslash}$ _enumext_start_item_viii:w.)

__enumext_stop_item_viii: The function __enumext_stop_item_viii: shall terminate with the capture of \item and its \(\chiotents \). Close the environments minipage, lrbox and the group. Then we only have to set the width of the box and print it next to \footnote, and add the horizontal and vertical separation between the boxes.

```
3933 \cs_new_protected_nopar:Nn \__enumext_stop_item_viii:
3934 {
©2024 by Pablo González L
```

```
\__enumext_endminipage:
          \end1rhox
        \group end:
        \box_set_wd:Nn \l__enumext_item_text_viii_box
3938
3939
            \l__enumext_joined_width_viii_dim
            + \l__enumext_labelwidth_viii_dim
3941
            + \l__enumext_labelsep_viii_dim
3942
          }
3943
        \int_set:Nn \hbadness { 10000 }
        \box_use:N \l__enumext_item_text_viii_box
        \bool_if:NF \l__enumext_footnotes_key_bool
              _enumext_print_footnote:
          }
        \int_compare:nNnTF { \l__enumext_item_column_pos_viii_int } = { \l__enumext_columns_viii_int
3950
          {
3951
            \par\noindent
3952
            \int_zero:N \l__enumext_item_column_pos_viii_int
3953
          { \hspace{ \l__enumext_columns_sep_viii_dim } }
3955
(End of definition for \__enumext_stop_item_viii:.)
```

__enumext_remove_extra_parsep_viii:

Finally we will remove the vertical space equal to \parsep when the total number of items is divisible by the number of items in the last row of the environment.

```
\cs_new_protected:Nn \__enumext_remove_extra_parsep_viii:
     {
       \int_compare:nNnT
3959
         {
3960
            \int_mod:nn { \g__enumext_item_count_all_viii_int } { \l__enumext_columns_viii_int }
3961
         }
3962
         =
3963
          {
           \c_zero_int }
         {
3965
            \par
            \vspace{ -\l__enumext_itemsep_viii_skip }
            \int_gzero:N \g__enumext_item_count_all_viii_int
3968
         }
3970
```

 $(\mathit{End of definition} \ for \ \verb|_-enumext_remove_extra_parsep_viii:.)$

11.36 The command \getkeyans

\getkeyans

The \getkeyans command takes a mandatory argument of the form $\{\langle store\ name: position \rangle\}$. Retrieve a "single" content stored by \anskey, \anspic* and \item* from $\langle prop\ list \rangle$ defined by save-ans key.

(End of definition for \getkeyans. This function is documented on page 13.)

__enumext_getkeyans_aux:n

The internal function $\ensuremath{\backslash}$ enumext_getkeyans_aux:n is in charge of *splitting* the $\ensuremath{\langle}$ argument $\ensuremath{\rangle}$ using ":". If ":" is omitted it will return an error.

(End of definition for $\ensuremath{\backslash}$ _enumext_getkeyans_aux:n.)

__enumext_getkeyans:nn

The internal function __enumext_getkeyans:nn will check for the existence of the $\langle prop \ list \rangle$, if it does not exist it will return an error message, then it will fetch the content specified by the second $\langle argument \rangle$ from $\langle prop \ list \rangle$.

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_getkeyans:nn.|)$

11.37 The command \printkeyans

The \printkeyans command prints "all stored content" in the $\langle sequence \rangle$ defined by the save-ans key. The first thing we will do is define a set of $\langle filtered\ keys \rangle$ with which we will control the options of the different nesting levels for the environment enumext and enumext* by storing their values in the list of tokens \l__enumext_print_keyans_X_tl. The variable \l__enumext_print_keyans_starred_tl will have the default $\langle keys \rangle$ for \printkeyans* and will be set by \setenumext[print*] and the variable \l__enumext_print_keyans_vii_tl will have the default keys for the environment enumext* nested within the $\langle sequence \rangle$ and will be set by \setenumext[print,*], the rest of the variables will be for the environment enumext and will be set by \setenumext[print, level].

The reason for storing \(\lambda \text{keys} \rangle \) in token lists using \(\text{keys_precompile:neN} \) is because the keys are set via \(\text{setenumext} \text{ but are later executed by running the command \\ \text{printkeyans} \) and they are not handled directly by its optional argument, except those related to the first opening level.

```
3997 \cs_generate_variant:Nn \keys_precompile:nnN { neN }
3998 \keys_define:nn { enumext / print }
    {
      print*
               .code:n
                           = \keys_precompile:neN { enumext / enumext* }
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_starred_tl, % starred cmd
       print* .initial:n = { nosep, label=\arabic*., columns=2, first=\small, font=\small },
      print-1 .code:n
                           = \keys_precompile:neN { enumext / level-1 }
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_i_tl,
      print-1 .initial:n = { nosep, label=\arabic*., columns=2, first=\small, font=\small },
       print-2 .code:n
                           = \keys_precompile:neN { enumext / level-2 }
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_ii_tl,
       print-2 .initial:n = { nosep, label=(\alph*), first=\small, font=\small },
      print-3 .code:n
                           = \keys_precompile:neN { enumext / level-3 }
4012
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_iii_tl,
      print-3 .initial:n = { nosep, label=\roman*., first=\small, font=\small },
                           = \keys_precompile:neN { enumext / level-4 }
      print-4 .code:n
                               { \__enumext_filter_save_key:n {#1} }
4017
                               \l__enumext_print_keyans_iv_tl,
4018
      print-4 .initial:n = { nosep, label=\Alph*., first=\small, font=\small },
4019
                           = \keys_precompile:neN { enumext / enumext* }
      print-* .code:n
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_vii_tl, % starred nested
       print-* .initial:n = { nosep, label=(\alph*), first=\small, font=\small },
    }
```

\printkeyans

Create a user command to print "all stored content" in $\langle sequence \rangle$ for \anskey, \item* and \anspic*. Within a group we will run our "precompiled keys" and then call the internal function __enumext_-printkeyans:nnn.

```
4025 \NewDocumentCommand \printkeyans { s O{} m }
4026 {
4027 \group_begin:
4028 \tl_use:N \l__enumext_print_keyans_i_tl
4029 \tl_use:N \l__enumext_print_keyans_ii_tl
4030 \tl_use:N \l__enumext_print_keyans_iii_tl
4031 \tl_use:N \l__enumext_print_keyans_iv_tl
4032 \tl_use:N \l__enumext_print_keyans_vii_tl
4033 \__enumext_printkeyans:nnn { #1 } { #2 } { #3 }
```

```
\group_end:
4035 }
```

#1: starred

(End of definition for \printkeyans. This function is documented on page 13.)

__enumext_printkeyans:nnn

The internal function __enumext_printkeyans:nnn will check for the existence of the $\langle sequence \rangle$, if it does not exist it will return an error message, then it will check the starred argument, if it is present it will execute the variable \l__enumext_print_keyans_starred_tl that contains the default $\langle keys \rangle$ for the environment enumext* not nested in the $\langle sequence \rangle$, it will open the environment enumext* passing the optional argument to the first level and then will map the $\langle sequence \rangle$, otherwise it will open the environment enumext passing the optional argument to the first level and then map the $\langle sequence \rangle$.

```
#2: key-val
#3: seq-name
4036 \cs_new_protected:Npn \__enumext_printkeyans:nnn #1 #2 #3
4037
       \seq_if_exist:cTF { g__enumext_#3_seq }
4038
4039
            \seq_if_empty:cF { g__enumext_#3_seq }
4040
4041
                %%\seq_show:c { g__enumext_#3_seq }
                \bool_if:nTF {#1}
                  {
                    \tl_use:N \l__enumext_print_keyans_starred_tl
                    \begin{enumext*}[#2]
                      \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
                    \end{enumext*}
4048
                  }
                  {
4050
                    \begin{enumext}[#2]
4051
                      \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
                    \end{enumext}
                  }
              }
         }
         {
4057
            \msg_error:nnn { enumext } { undefined-storage-anskey } {#3}
4058
         }
4059
4060
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_-enumext_printkeyans:nnn.|)$

11.38 The command \setenumext

First we define a "meta families" of $\langle keys \rangle$ to access from \setenumext.

```
4061 \keys_define:nn { enumext / meta-families }
4062
      enumext-1 .code:n = { \keys_set:nn { enumext / level-1 } {#1} } ,
4063
      enumext-2 .code:n = { \keys_set:nn { enumext / level-2 } {#1} } ,
4064
      enumext-3 .code:n = { \keys_set:nn { enumext / level-3 } {#1} } ,
      enumext-4 .code:n = { \keys_set:nn { enumext / level-4 } {#1} } ,
4066
                .code:n = { \keys_set:nn { enumext / keyans } {#1} } ,
      enumext* .code:n = { \keys_set:nn { enumext / enumext* } {#1} } ,
      keyans* .code:n = { \keys_set:nn { enumext / keyans* } {#1} } ,
      print*
                .code:n = { \keys_set:nn { enumext / print } { print* = {#1} } } ,
      print-1
               .code:n = { \keys_set:nn { enumext / print
                                                              } { print-1 = {#1} } } ,
      print-2 .code:n = { \keys_set:nn { enumext / print
                                                               } { print-2 = {#1} } } ,
      print-3
                .code:n = { \keys_set:nn { enumext / print
                                                               } { print-3 = {#1} } } ,
4073
      print-4
                 .code:n = { \keys set:nn { enumext / print
                                                               } { print-4 = {#1} } } ,
4074
      print-*
                 .code:n = { \keys_set:nn { enumext / print
                                                               } { print-* = {#1} } } ,
4075
      unknown
                 .code:n = { \msg_error:nn { enumext } { unknown-key-family } } ,
4076
4077
```

We store them in the constant sequence \c__enumext_all_families_seq separated by commas.

```
4078 \seq_const_from_clist:Nn \c__enumext_all_families_seq
4079 {
4080 enumext-1, enumext-2, enumext-3, enumext-4, keyans, enumext*,
4081 keyans*, print-1, print-2, print-3, print-4, print-*, print*,
4082 }
```

\setenumext Now we define the user command \setenumext.

```
4083 \NewDocumentCommand \setenumext { O{enumext,1} +m }
       \tl_if_novalue:nTF {#1}
4085
4086
         {
           \seq_map_inline:Nn \c__enumext_all_families_seq
4087
         }
4088
         {
           \seq_clear:N \l__enumext_setkey_tmpa_seq
           \seq_set_from_clist:Nn \l__enumext_setkey_tmpb_seq {#1}
4091
           \int_set:Nn \l__enumext_setkey_tmpa_int
               \seq_count:N \l__enumext_setkey_tmpb_seq
             7
           \int_compare:nNnTF { \l__enumext_setkey_tmpa_int } > { 1 }
             {
               \seq_pop_left:NN \l__enumext_setkey_tmpb_seq \l__enumext_setkey_tmpa_tl
4098
               \seq_map_function:NN \l__enumext_setkey_tmpb_seq \__enumext_set_parse:n
               \seq_set_map_e:NNn \l__enumext_setkey_tmpa_seq \l__enumext_setkey_tmpa_seq
                    \tl_use:N \l__enumext_setkey_tmpa_tl - ##1
                  }
             }
             {
               \seq_put_right:Ne \l__enumext_setkey_tmpa_seq { \tl_trim_spaces:n {#1} }
4107
           \seq_if_empty:NTF \l__enumext_setkey_tmpa_seq
4108
             { \seq_map_inline:Nn \c__enumext_all_families_seq }
4109
              { \seq_map_inline:Nn \l__enumext_setkey_tmpa_seq }
4110
         }
4111
4112
           \keys_set:nn { enumext / meta-families } { ##1 = {#2} }
4113
4115
```

 $(\textit{End of definition for } \backslash \textit{setenumext}. \ \textit{This function is documented on page 6.})$

__enumext_set_parse:n
__enumext_set_error:nn

Internal functions used by the \setenumext command.

```
4116 \cs_new_protected:Npn \__enumext_set_parse:n #1
4117
       \tl_set:Ne \l__enumext_setkey_tmpb_tl { \tl_trim_spaces:n {#1} }
4118
       \clist_map_inline:nn { 0, 1, 2, 3, 4, * } % <- max level
4119
         { \tl_remove_all:Nn \l__enumext_setkey_tmpb_tl {##1} }
4120
       \tl_if_empty:NTF \l__enumext_setkey_tmpb_tl
4121
4122
            \seq_put_right:Ne \l__enumext_setkey_tmpa_seq
4123
              { \tl_trim_spaces:n {#1} }
4124
4125
         { \__enumext_set_error:nn {#1} { } }
_{\mbox{\tiny 4128}} \cs_new_protected:Npn \__enumext_set_error:nn #1 #2
     { \msg_error:nnn { enumext } { invalid-key } {#1} {#2} }
```

(End of definition for $\ _$ enumext_set_parse:n and $\ _$ enumext_set_error:nn.)

11.39 Messages

Message used by package-load for ${\tt multicol}$ and ${\tt hyperref}$ packages.

Message used in the creation of counters by enumext package.

```
4142 \msg_new:nnn { enumext } { counters }
       The ~ counter ~ '#1' ~ is ~ already ~ defined ~ by ~ some ~ \
4144
       package ~ or ~ macro, ~ it ~ cannot ~ be ~ continued.
4145
4146
Message used in the creation of \langle prop \ list \rangle by enumext package.
4147 \msg_new:nnn { enumext } { store-prop }
         ~ Package ~ enumext: ~ Creating ~ \c_backslash_str g__enumext_#1_prop ~ \msg_line_context:.
4149
4150
4151 \msg_new:nnn { enumext } { store-seq }
4152
         ~ Package ~ enumext: ~ Creating ~ \c_backslash_str g__enumext_#1_seq ~ \msg_line_context:.
   \msg_new:nnn { enumext } { store-int }
4155
         ^{\sim} Package ^{\sim} enumext: ^{\sim} Creating ^{\sim} \c_backslash_str g__enumext_resume_#1_int ^{\sim} \msg_line_con
4158
   \msg_new:nnn { enumext } { prop-seq-int-hook }
4160
       * ~ Package ~ enumext: ~ Elements ~ in ~ \c_backslash_str g__enumext_#1_prop ~ = ~ #2.\\
4161
       * ~ Package ~ enumext: ~ Elements ~ in ~ \c_backslash_str g__enumext_#1_seq ~ = ~ #3.\\
       * ~ Package ~ enumext: ~ Value ~ off ~ \c_backslash_str g__enumext_resume_#1_int ~ = ~ #4.
4164
4165 \msg_new:nnn { enumext } { item-answer-hook }
        * ~ Package ~ enumext: ~ Value ~ off ~ \c_backslash_str g__enumext_item_number_int ~ = ~ #1.\
       * ~ Package ~ enumext: ~ Value ~ off ~ \c_backslash_str g__enumext_item_anskey_int ~ = ~ #2.\
        * ~ Package ~ enumext: ~ Difference ~ item_number_int ~ - ~ item_anskey_int ~ = ~ #3.
4170
Message used by \lceil \langle key = val \rangle \rceil system and \setenumext command.
4171 \msg_new:nnn { enumext } { invalid-key }
       The ~ kev ~ '#1' ~ is ~ not ~ know ~ the ~ level ~ #2.
4173
4174
4175 \msg_new:nnn { enumext } { unknown-key-family }
4176
       Unknown~key~family~`\l_keys_key_str'~for~enumext.
4177
4178
Messages used in length calculation.
4179 \msg_new:nnn { enumext } { width-negative }
4180
       Ignoring ~ negative ~ value ~ '#1=#2' ~ \msg_line_context:.\\
4181
       The \sim key \sim '#1'\sim accepts \sim values \sim >= \sim Opt.
4182
4183
4184 \msg_new:nnn { enumext } { width-zero }
4185
       Invalid ~ '#1=#2' ~ \msg_line_context:.\\
       The ~ key ~ '#1'~ accepts ~ values ~ > ~ Opt.
4187
4188
Messages used by show-length key in enumext.
4189 \msg_new:nnn { enumext } { list-lengths }
        **** ~ Lengths ~ used ~ by ~ 'enumext' ~ level ~ '#2' ~ \msg_line_context:~\c_space_tl ****\\
          _enumext_show_length:nnn { dim } { labelsep
       \__enumext_show_length:nnn { dim } { labelwidth
                                                               } {#1}
       \__enumext_show_length:nnn { dim } { itemindent } {#1}
4194
       \__enumext_show_length:nnn { dim } { leftmargin } {#1}
       \__enumext_show_length:nnn { dim } { rightmargin } {#1}
4196
        \__enumext_show_length:nnn { dim } { listparindent } {#1}
4197
       \__enumext_show_length:nnn { skip } { topsep } {#1}
       \__enumext_show_length:nnn { skip } { parsep
       \__enumext_show_length:nnn { skip } { partopsep } {#1}
       \__enumext_show_length:nnn { skip } { itemsep } {#1}
```

```
Messages used by show-length key in enumext*, keyans* and keyans.
4204 \msg_new:nnn { enumext } { list-lengths-not-nested }
       **** ~ Lengths ~ used ~ by ~ '#2' ~ environment ~ \msg_line_context:~\c_space_tl ****\\
4206
       \__enumext_show_length:nnn { dim } { labelsep
                                                           } {#1}
4207
       \__enumext_show_length:nnn { dim } { labelwidth
                                                            } {#1}
4208
       \__enumext_show_length:nnn { dim } { itemindent
                                                            } {#1}
       \__enumext_show_length:nnn { dim } { leftmargin
                                                            } {#1}
4210
       \__enumext_show_length:nnn { dim } { rightmargin
4211
       \__enumext_show_length:nnn { dim } { listparindent } {#1}
       \__enumext_show_length:nnn { skip } { topsep
       \__enumext_show_length:nnn { skip } { parsep
       \__enumext_show_length:nnn { skip } { partopsep } {#1}
       \__enumext_show_length:nnn { skip } { itemsep } {#1}
4216
4217
    }
4218
Messages used by ref key.
4219 \msg_new:nnn { enumext } { key-ref-empty }
       Key ~ 'ref' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
4222
Messages used by save-ans key.
4223 \msg_new:nnn { enumext } { save-ans-empty }
       Key ~ 'save-ans' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
4225
4227 \msg_new:nnn { enumext } { save-ans-log }
       * ~ Package ~ enumext: ~ Start ~ \c_left_brace_str#1\c_right_brace_str \c_space_tl with ~ sav
   ans=#2 ~ \msg_line_context:.
    }
4231 \msg_new:nnn { enumext } { save-ans-log-hook }
4232
       * ~ Package ~ enumext: ~ Stop ~ \c_left_brace_str#1\c_right_brace_str \c_space_tl with ~ save
   ans=#2 ~ \msg_line_context:.
4234
4235 \msg_new:nnn { enumext } { save-ans-hook }
4236
       Stop ~ storing ~ for ~ 'save-ans=#1' ~ \msg_line_context:.
4237
Messages used by the internal system to check answer used by check-ans key.
4239 \msg_new:nnn { enumext } { need-save-ans }
       Key ~ '#1'~ works ~ only ~ with ~ the ~ 'save-ans' ~ key ~ in ~ '#2'~ \msg_line_context:.
4243 \msg_new:nnn { enumext } { items-same-answer }
       ***********
       * ~ Package ~ enumext: ~ Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right
4246
       * ~ started ~ #3 ~ and ~ close ~ \msg_line_context: : ~ 'OK', ~ all ~ items ~ with ~ answer.\
4247
4248
4250 \msg_new:nnn { enumext } { item-greater-answer }
       Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\\
       started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\\
       Items ~ > ~ Answers.
4255
4256 \msg_new:nnn { enumext } { item-less-answer }
4257
       Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\\
4258
       started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\\
4259
       Items ~ < ~ Answers.
Messages used by the internal system to check for "starred" \item* and \anspic* commands.
4262 \msg_new:nnn { enumext } { missing-starred }
       Missing ~ '\c_backslash_str #1*' ~ #2.
```

```
4266 \msg_new:nnn { enumext } { many-starred }
       Many ~ '\c_backslash_str #1*' ~ #2.
4268
Message for the nesting depth of the environment {\tt enumext}.
4270 \msg_new:nnn { enumext } { list-too-deep }
       Too ~ deep ~ nesting ~ for ~ 'enumext' ~ \msg_line_context:.~ \\
4272
       The ~ maximum ~ level ~ of ~ nesting ~ is ~ 4.
4274
Messages used by \anskey and \anspic commands.
4275 \msg_new:nnn { enumext } { anskey-wrong-place }
       Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:.~ \\
4277
       '\c_backslash_str #1' ~ works ~ in ~ the ~ environment ~ '#2'.
4279
4280 \msg_new:nnn { enumext } { anspic-wrong-place }
       Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:.~ \\
       '\c_backslash_str #1' ~ works ~ in ~ the ~ environment ~ '#2'.
4285 \msg_new:nnn { enumext } { command-wrong-place }
       Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:.~ \\
4287
       '\c_backslash_str #1' ~ works ~ outside ~ the ~ environment ~ '#2'.
4288
Messages used by keyans and keyanspic environment.
4290 \msg_new:nnn { enumext } { keyans-nested }
       The ~ environment ~ 'keyans' ~ can't ~ be ~ nested ~ \msg_line_context:.
4292
4294 \msg_new:nnn { enumext } { keyans-wrong-level }
4295
       Wrong ~ level ~ position ~ for ~ 'keyans' ~ \msg_line_context:.~ \\
4296
       The ~ environment ~ 'keyans' ~ can ~ only ~ be ~ in ~ the ~ first ~ level.
4297
4299 \msg_new:nnn { enumext } { wrong-place }
       Wrong ~ place ~ for ~ '#1' ~ environment ~\msg_line_context:.~ \\
       '#1' ~ is ~ only ~ found ~ with ~ '#2' ~ in ~ 'enumext.
4304 \msg_new:nnn { enumext } { keyanspic-nested }
       The ~ environment ~ 'keyanspic' ~ can't ~ be ~ nested~ \msg_line_context:.~.
4306
4307
4308 \msg_new:nnn { enumext } { keyanspic-wrong-level }
       Wrong ~ level ~ position ~ for ~ 'keyanspic' ~ \msg_line_context:.~ \\
       The ~ environment ~ 'keyans' ~ can ~ only ~ be ~ in ~ the ~ first ~ level.
Messages used by \getkeyans command.
4313 \msg_new:nnn { enumext } { undefined-storage-anskey }
       Storage ~ named ~ '#1' ~ is ~ not ~ defined ~ \msg_line_context:.
Messages used by \miniright command.
4317 \msg_new:nnn { enumext } { missing-miniright }
       Missing ~ '\c_backslash_str miniright' ~ in ~ \msg_line_context:.\\
       The ~ key ~ 'mini-env' ~ need ~ '\c_backslash_str miniright'.
4321
4322 \msg_new:nnn { enumext } { wrong-miniright-place }
4323
       Wrong ~ place ~ for ~ '\c_backslash_str miniright' ~ \msg_line_context:.~ \\
4324
       Works ~ in ~ 'enumext' ~ and ~ 'keyans' ~ with ~ key ~ 'mini-env'.
4325
4326
4327 \msg_new:nnn { enumext } { wrong-miniright-use }
```

11.40 Finish package

Finish package implementation.

```
_{4344} \file_input_stop: _{4345} \langle /package \rangle
```

12 Index of Implementation

The italic numbers denote the pages where the corresponding entry is described, the numbers underlined and all others indicate the line on which they are implemented in the package code.

Symbols	\bool_not_p:n 218, 227, 2226, 2318, 2333, 2914, 2915,
* 201	2927
\+ 193	\bool_set_eq:NN 2639, 2687, 3484, 3817
\ 193	\bool_set_false:N 368, 1791, 1792, 3027, 3059, 3141,
\\ 209, 3167, 4144, 4161, 4162, 4167, 4168, 4181, 4186, 4191,	3206, 3224, 3436, 3481, 3767, 3814
4206, 4245, 4246, 4247, 4252, 4253, 4258, 4259, 4272,	\bool_set_true:N 244, 258, 350, 354, 460, 778, 1371,
4277, 4282, 4287, 4296, 4301, 4310, 4319, 4324, 4329	1376, 1642, 1764, 1765, 2044, 2052, 2635, 2665, 2683, 2695, 2889, 2920, 2933, 2959, 3056, 3083, 3341, 3410,
A	3490, 3497, 3498, 3699, 3823, 3830, 3831
above	box commands:
above*	\box_dp:N 1059, 1063, 1067, 1078, 1082, 1093, 1102,
\addvspace 1012, 1040, 1163, 1242, 1305, 1311, 1339, 1356,	1108, 1118, 1131, 1137, 1143, 1174, 1175, 1176, 1179,
3006, 3021, 3123, 3138, 3364, 3371, 3722, 3729	1189, 1193, 1202, 1209, 1214, 1222, 1251, 1252, 1255,
after <u>850</u>	1262, 1275, 1283, 1289, 1297, 3236
align 467	\box_new:N 58, 162
\Alph 32, 37	\box_set_wd:Nn 3587, 3938
\Alph 419, 528, 573, 641, 4019	\box_use:N 3594, 3945
\alph 32, 37	\box_wd:N 426
\alph 420, 526, 4011, 4023	
\anskey	C
\anspic	\c 201, 202, 678, 680, 692, 694
\anspic* 63	\cB
\arabic 27, 32	\cE
\arabic 418, 525, 572, 4003, 4007	check-ans
В	Document class:
\baselineskip 44	article
\baselineskip	clist commands:
before 850	\clist_const:Nn 174
before* 850	\clist_map_function:nN 3244
below	\clist_map_inline:Nn . 466, 720, 783, 849, 864, 945,
below* 1365	1381
bool commands:	\clist_map_inline:nn 36, 47, 66, 72, 84, 96, 120, 150,
\bool_gset_false:N 307, 308, 309, 3373, 3377, 3731	173, 491, 508, 788, 960, 1487, 1731, 1797, 1991, 2009,
\bool_gset_true:N 221, 230, 954, 1856, 1862, 3356,	2041, 2288, 2422, 2589, 2800, 2803, 2830, 2840, 2843,
3374, 3714, 3732	2863, 4119
\bool_if:NTF . 359, 371, 388, 1387, 1401, 1414, 1425,	\columnbreak
1436, 1447, 1458, 1469, 1522, 1539, 1544, 1552, 1579,	\columnbreak
1617, 1622, 1629, 1633, 1655, 1660, 1668, 1675, 1706,	columns-sep
1714, 1806, 1930, 2012, 2022, 2101, 2125, 2132, 2160,	\columnsep
2191, 2204, 2214, 2234, 2359, 2370, 2374, 2413, 2428, 2503, 2514, 2518, 2631, 2661, 2735, 2751, 2813, 2823,	\columnsep
2853, 2858, 2940, 2990, 3004, 3012, 3051, 3108, 3121,	\columnseprule
3129, 3147, 3352, 3361, 3365, 3441, 3451, 3534, 3539,	\columnseprule 2988, 3107
3547, 3551, 3566, 3595, 3710, 3719, 3723, 3863, 3867,	Commands provide by enumext:
3891, 3900, 3904, 3910, 3924, 3946	\anskey 24, 25, 59, 60, 65, 66, 68, 70-72, 74, 83, 95, 105,
\bool_if:nTF 1340, 1357, 2242, 2672, 2707, 2771, 3168,	106, 111
4043	\anspic* 24, 25, 63, 72, 73, 89-91, 105, 106
\bool_if_p:N . 239, 253, 1686, 1687, 1695, 1696, 1819,	\anspic
1841, 1853, 1854, 1859, 1860, 2225, 2268, 2269, 2293,	\getkeyans
2302, 2303, 2315, 2331, 2490, 2491, 2528, 2529, 2913,	\item* 24, 25, 63, 66, 72, 73, 77, 78, 96, 103, 105, 106
2926, 2928, 3175, 3176	\itemwidth 91, 92, 99
\bool_lazy_all:nTF 237, 251, 1817, 1839, 2291, 2300,	\item 76, 78, 92, 95–97, 99, 102
2313, 2329, 2911, 2924	\miniright 24, 42, 50, 51, 84, 85, 87, 88, 111
\bool_lazy_and:nnTF 217, 226, 1685, 1694, 1852,	\printkeyans*
1858, 2224, 2267, 2489 \bool_lazy_or:nnTF 1747, 1754, 2527, 3174	\setenumext
\bool_new:N 25, 26, 27, 28, 29, 30, 31, 51, 61, 82, 87, 88,	Counters defined by enumext:
93, 94, 97, 115, 118, 121, 122, 131, 132, 133, 141, 142,	enumXiii
156, 167, 169	enumXii

enumXiv 23, 32	3097, 3249, 3267, 3274, 3317, 3335, 3522, 3625, 3632
enumXi 23, 32	3675, 3693
enumXviii 23, 32	\dim_set_eq:NN 516, 563, 634, 638, 2654, 2802, 2842
enumXvii 23, 32, 97	2984, 3105, 3324, 3327, 3328, 3513, 3682, 3685, 3686
enumXvi 23, 32	\dim_use:N 792, 800, 1332, 1338, 2168, 2171, 2176, 2724
enumXv	2726, 2951, 2956, 2957, 2964, 2974, 2978, 2979, 2981
cs commands:	\dim_zero:N 2988, 3107, 3228, 3229, 3230
\cs_generate_variant:Nn 428, 444, 684, 700, 2093,	\dim_zero_new:N 3280, 3638
2098, 2178, 2790, 3246, 3997	\c_zero_dim 794, 808, 820, 832, 1332, 1350, 2254, 2761
\cs_if_exist:NTF 398	2766, 2772, 2779, 2951, 2974, 3077, 3095, 3265, 3333
\cs_new:Nn 187	3623, 3691
\cs_new:Npn . 205, 1488, 1497, 1506, 2056, 2065, 2073	
\cs_new_eq:NN 334, 335, 336, 340, 341, 373, 374, 377,	E
378	\end 1335, 1353, 2127, 2162, 3003, 3020, 3120, 3137, 3354
\cs_new_protected:Nn . 197, 211, 235, 266, 293, 299,	3370, 3712, 3728, 4048, 4053
305, 311, 317, 325, 345, 549, 612, 664, 865, 869, 873,	\endlist 30
877, 881, 885, 889, 893, 897, 901, 905, 909, 913, 917,	\endlist 335
921, 925, 961, 973, 997, 1014, 1025, 1049, 1124, 1148,	\endlrbox 3585, 3936
1165, 1227, 1244, 1266, 1301, 1307, 1382, 1396, 1410,	\endminipage 30
1421, 1432, 1443, 1454, 1465, 1550, 1653, 1666, 1683,	\endminipage 34
1704, 1732, 1737, 1762, 1802, 1812, 1850, 1865, 1872,	enumext
1881, 1886, 1891, 1896, 1905, 1910, 1915, 1920, 2099,	enumext internal commands:
2123, 2130, 2158, 2165, 2279, 2411, 2426, 2454, 2487,	\lenumext⊔_check_start_line_env_tl 20
2523, 2535, 2543, 2594, 2598, 2617, 2668, 2703, 2719,	\lambda_enumext $_{\sqcup}$ ref_the_count_tl 32
2729, 2745, 2883, 2909, 2938, 2945, 2968, 2998, 3010,	\l_enumext_resume_name_tl 55
3049, 3073, 3091, 3116, 3127, 3164, 3208, 3222, 3242,	\enumext_add_pre_parsep: 43, 971, 973, 973
3247, 3263, 3331, 3350, 3402, 3423, 3430, 3439, 3449,	\enumext_after_args_exec: . 41, 865, 877, 2876
3466, 3606, 3621, 3689, 3708, 3758, 3780, 3786, 3799,	\enumext_after_args_exec_v: 41, 42, 881, 893
3855, 3957	3042
\cs_new_protected:Npn 179, 183, 381, 396, 413, 423,	\enumext_after_args_exec_vii: 897, 923
429, 529, 574, 646, 671, 685, 1329, 1348, 1518, 1537,	
1607, 1640, 1742, 1939, 2010, 2020, 2042, 2050, 2085,	\enumext_after_args_exec_viii: 925
2094, 2211, 2356, 2368, 2390, 2464, 2508, 2627, 2645,	\enumext_after_env:nn 62, 85, 98, <u>183</u> , 183, 3030
2679, 2691, 2759, 2793, 2833, 2892, 3069, 3217, 3282,	3359, 3620, 3717
3413, 3472, 3479, 3495, 3503, 3508, 3520, 3640, 3773,	\enumext_after_hyperref: 30, 343, 345, 345
3805, 3812, 3828, 3836, 3850, 3976, 3989, 4036, 4116,	\enumext_after_list: 85, 95, 101, 2881, <u>3</u> 010
4128	3010
\cs_new_protected_nopar:Nn 3459, 3582, 3792,	\lenumext_after_list_args_v_tl 890
3933	\lenumext_after_list_args_vii_tl 923, 3576
\cs_new_protected_nopar:Npn 3526, 3883	\lenumext_after_list_args_viii_tl 927, 3920
\cs_set:Nn 2361	\enumext_after_list_v: 88, 3047, <u>3127</u> , 3127
\cs_set:Npn 2289, 2327, 3982	\enumext_after_list_vii: 3400, <u>3430</u> , 3430
\cs_set_eq:NN 3392, 3393, 3528, 3748, 3749, 3885	\enumext_after_list_viii: 3756, <u>3786</u> , 3786
\cs_set_protected:Nn 789, 805, 817, 829	\enumext_after_star_env:nn 9
\cs_set_protected:Npn . 32, 41, 59, 67, 79, 85, 112,	\enumext_after_stop_list: 41, 42, <u>865</u> , 873
146, 154, 445, 467, 496, 509, 556, 701, 721, 765, 784,	3025
841, 850, 929, 946, 1365, 1476, 1723, 1783, 1956, 1992,	\enumext_after_stop_list_v: 41, <u>881</u> , 889, 3142
2028, 2281, 2415, 2578, 2791, 2831	\lenumext_after_stop_list_v_tl 89:
\cs_to_str:N 415, 438	\enumext_after_stop_list_vii: 897,913,3433
	\l_enumext_after_stop_list_vii_tl 91
D	\enumext_after_stop_list_viii: . 917,3789
\d 193	\l_enumext_after_stop_list_viii_tl 919
\DeclareDocumentEnvironment 1042	\l_enumext_align_label_vii_str 3568, 3572
dim commands:	\l_enumext_align_label_viii_str . 3912, 3910
\dim_abs:n 2764, 2769	\l_enumext_align_label_X_str 154
\dim_add:Nn 3227	\cenumext_all_envs_clist 174, 466, 720, 783
\dim_compare:nNnTF . 791, 807, 819, 831, 1331, 1350,	849, 864, 945, 1381
2761, 2766, 2772, 2778, 2780, 2782, 2950, 2973, 3077,	\cenumext_all_families_seq 107, 4078, 4087
3095, 3219, 3265, 3333, 3623, 3691	4109
\dim_compare:nTF 2252	\enumext_anskey_wrapper:n 1960, 2360
\dim_gset_eq:NN	\enumext_at_begin_document:n 30, 179, 179
\dim_gzero:N	332, 338
\dim_new:N 55, 62, 63, 64, 81, 127, 163, 164, 170	\enumext_before_args_exec: 40,865,865,294
\dim_set:Nn 426, 779, 2659, 2764, 2769, 2771, 2774,	\enumext_before_args_exec_v: 41, 881, 881
2775, 2779, 2781, 2784, 2785, 2787, 2953, 2976, 3079,	3076

114/125

\enumext_before_args_exec_vii: <u>897</u> , 897,
34 ² 7 \enumext_before_args_exec_viii: 901,3783
\enumext_before_keys_exec: 40,865,869,2874
\enumext_before_keys_exec_v: 41, 881, 885,
3040
\enumext_before_keys_exec_vii 897
\enumext_before_keys_exec_vii: 41,905,3388
_enumext_before_keys_exec_viii: 41,909,
3744
\enumext_before_list: 84, 2868, 2945, 2945
\enumext_before_list_v: . 86, 3035, 3073, 3073
\enumext_before_list_vii: 94, 3383, 3423, 3423
\enumext_before_list_viii: 101, 3740, <u>3780</u> ,
3780
\lenumext_before_no_starred_key_v_tl 887
\lenumext_before_no_starred_key_vii
tl 907 \lenumext_before_no_starred_key_viii
tl
\lenumext_before_starred_key_v_tl 883
\lenumext_before_starred_key_vii_tl . 899
\lenumext_before_starred_key_viii_tl 903
\enumext_calc_hspace:NNNNNNN 80, 2759, 2759,
2790, 2795, 2835
\enumext_check_ans_active: $60, 84, \underline{1802}, 1802,$
2949, 3426
\genumext_check_ans_item_tl 74
\genumext_check_ans_key_bool 61, 62, 131, 307,
1856, 1862, 1930
\lenumext_check_ans_key_bool 61, 76, 77, <u>131</u> , 1787, 1792, 1853, 1859
_enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434
\enumext_check_ans_key_hook: 61, <u>1850</u> , 1850,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 _enumext_check_ans_show: 61, 62, 1872, 1872, 1932
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_ans_show_bool 85 \lenumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_ans_show_bool 85 \lenumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1876, 1881 _enumext_check_ans_msg_same_ok: 1872, 1877, 1886 _enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \l_enumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 _enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \l_enumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 _enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \g_enumext_check_starred_cmd_int 131, 1942,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \l_enumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 _enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \g_enumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 \enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \genumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \lenumext_check_starrel_line_env_tl 131, 272,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_answers_bool 85 \lenumext_check_answers_bool 85 \lenumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 \enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \genumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_start_line_env_tl 131, 272, 279, 286, 1945, 1951, 1954
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_ans_show_bool 85 \l_enumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 \enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \genumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_lenv_tl 131, 272, 279, 286, 1945, 1951, 1954 \l_enumext_check_starred_vdim 3095, 3097, 3105
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1876, 1881 _enumext_check_ans_msg_same_ok: 1872, 1877, 1886 _enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_ans_show_bool 85 \l_enumext_check_ans_show_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 _enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \genumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_cmd_int 131, 272, 279, 286, 1945, 1951, 1954 \l_enumext_columns_sep_vdim 3095, 3097, 3105 \l_enumext_columns_sep_vdidm 3265, 3267,
\enumext_check_ans_key_hook: 61, 1850, 1850, 3024, 3434 \enumext_check_ans_level: 60, 1802, 1808, 1812 \enumext_check_ans_log: 61, 62, 1896, 1896, 1934 \enumext_check_ans_log_msg_greater: 1896, 1902, 1915 \enumext_check_ans_log_msg_less: 1896, 1900, 1905 \enumext_check_ans_log_msg_same_ok: 1896, 1901, 1910 \enumext_check_ans_msg_greater: 1872, 1878, 1891 \enumext_check_ans_msg_less: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1876, 1881 \enumext_check_ans_msg_same_ok: 1872, 1877, 1886 \enumext_check_ans_show: 61, 62, 1872, 1872, 1932 \genumext_check_ans_show_bool 85 \lenumext_check_ans_show_bool 85 \l_enumext_check_answers_bool 59, 60, 131, 1765, 1791, 1806, 2101, 2125, 2132, 2160, 2204, 2503, 2631, 2661, 3539 \enumext_check_starred_cmd:n 28, 63, 74, 1939, 1939, 3045, 3203, 3754 \genumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_cmd_int 131, 1942, 1948, 1953, 2701, 3173, 3862 \l_enumext_check_starred_lenv_tl 131, 272, 279, 286, 1945, 1951, 1954 \l_enumext_check_starred_vdim 3095, 3097, 3105

```
\l__enumext_columns_v_int 1170, 3093, 3101, 3113,
    3118
\l__enumext_columns_vii_int . . 3270, 3273, 3277,
    3285, 3289, 3292, 3298, 3304, 3308, 3599, 3610
\l__enumext_columns_viii_int . 3628, 3631, 3635,
    3643, 3647, 3650, 3656, 3662, 3666, 3950, 3961
\g__enumext_count_item_with_ans_int .... 68
\l__enumext_counter_i_tl .... 32, 405
\l__enumext_counter_ii_tl ..... 32, 406
\l__enumext_counter_iii_tl .... 32, 407
\l__enumext_counter_iv_tl ..... 32, 408
\c__enumext_counter_style_tl .... 27, 37, 199
\g__enumext_counter_styles_tl . 23, 32, 55, 416,
\l__enumext_counter_v_tl .... 32, 409, 654
\l__enumext_counter_vi_tl ..... 32, 410
\l__enumext_counter_vii_tl . . . . . 32, 411, 584
\l__enumext_counter_viii_tl .... 32, 412, 601
\l__enumext_current_widest_dim 23, 55, 440, 517,
    564, 635, 639
\__enumext_default_item:n ... 2627, 2627, 2676
\__enumext_define_counters:Nn 23, 396, 396, 405,
    406, 407, 408, 409, 410, 411, 412
\__enumext_endminipage: 30, 338, 341, 1048, 3259,
    3584, 3935
\g__enumext_envir_name_tl 28, 136, 245, 259, 315,
    1735, 1740, 1750, 1884, 1889, 1894, 1908, 1913, 1918
\__enumext_execute_after_env: .. 29, 58, 61, 62,
    1920, 1920, 3030, 3620
\__enumext_fake_item: ..... 789, 789, 2822
\l__enumext_fake_item_indent_v_dim 808,813
\l__enumext_fake_item_indent_v_tl 810, 2684,
    2688, 2696
\l__enumext_fake_item_indent_vii_dim 820,825
\l__enumext_fake_item_indent_vii_tl 822, 3580
\l__enumext_fake_item_indent_viii_dim . 832,
    837, 3927
\l__enumext_fake_item_indent_viii_tl .. 834,
    3926, 3930
\l__enumext_fake_item_indent_X_tl .... 85
\__enumext_fake_item_vii: .... 789, 817, 2852
\__enumext_fake_item_viii: . . . . 789, 829, 2857
\__enumext_filter_save_key:n . . 65, 2017, 2025,
    2048, 2054, 2056, 2056, 4001, 4005, 4009, 4013, 4017,
    4021
\__enumext_filter_save_key_key:n .. 65, 2056,
    2061, 2065
\__enumext_filter_save_key_pair:nn
                                        66, 2056,
    2062, 2073
\__enumext_filter_series:n 53, 1488, 1488, 1530,
    1542, 1547
\__enumext_filter_series_key:n 54, 1488, 1493,
\__enumext_filter_series_pair:nn .. 54, 1488,
\g_{enumext\_footnote\_arg\_seq} . 151, 2600, 2613,
\g__enumext_footnote_int . 151, 2607, 2610, 2612,
\g_{\text{enumext\_footnote\_int\_seq}} . \underline{151}, \underline{2601}, \underline{2614},
    2619, 2622
\__enumext_footnotes_key_bool ..... 30
\l__enumext_footnotes_key_bool 25, 30, 97, 141,
    354, 359, 368, 3547, 3595, 3900, 3946
```

- $\verb|__enumext_footnotetext:nn| \dots \underline{2594}, 2594, 2624$ __enumext_getkeyans:nn . . 106, 3985, 3989, 3989 __enumext_getkeyans_aux:n 105, 3973, 3976, 3976 \l__enumext_hyperref_bool . 25, 30, 31, 141, 350, 371, 388, 2269, 2491, 3534, 3891 $\verb|__enumext_hypertarget:nn| 31, \underline{345}, 373, 377, 393$ __enumext_if_is_int:n 191 __enumext_if_is_int:nTF 191, 673, 687 __enumext_is_not_nested: 27, 82, 211, 211, 2885, 3404 __enumext_is_on_first_level: . 28, 82, 211, 235, 2890, 3411 \g__enumext_item_anskey_int . . 74, 131, 302, 329, 330, 1869, 2206, 2505 __enumext_item_answer_diff: 61, 62, 1865, 1865, \g__enumext_item_answer_diff_int . 61, 62, 140, 303, 1867, 1874, 1898 \l__enumext_item_column_pos_vii_int 95, 3292, 3298, 3304, 3308, 3315, 3462, 3599, 3602 \l__enumext_item_column_pos_viii_int .. 102, 3650, 3656, 3662, 3666, 3673, 3795, 3950, 3953 $\label{local_local_local} $$l_=numext_item_column_pos_X_int \dots 154$$ \g__enumext_item_count_all_vii_int 95, 3316, 3463, 3610, 3617 \g__enumext_item_count_all_viii_int 102, 3674, 3796, 3961, 3968 \g__enumext_item_count_all_X_int 154 $\verb|\g_enumext_item_number_int| \textit{ 60, 61, } 131, 301, 328,$ 330, 1823, 1827, 1830, 1833, 1845, 1869, 2633, 2663, 3541 __enumext_item_peek_args_vii: 95, 3464, 3466, 3466 __enumext_item_peek_args_viii: .. 102, 3797, 3799, 3799 __enumext_item_starred: .. 79, 2719, 2719, 2737 \l__enumext_item_starred_vii_bool 3481, 3497, \l__enumext_item_starred_viii_bool 3814, 3830, 3904, 3924 \l__enumext_item_starred_X_bool 154 __enumext_item_std:w 30, 76-78, 91, 332, 336, 2636, 2642, 2666, 2684, 2688, 2696, 3240 \g__enumext_item_symbol_aux_vii_tl 3505, 3553, 3556, 3560, 3562 \g__enumext_item_symbol_aux_X_tl 154 \l__enumext_item_symbol_sep_vii_dim .. 3514, 3522, 3559, 3561 \g__enumext_item_symbol_tl 23, 77, 48, 2651, 2725, \l__enumext_item_symbol_vii_tl 3556 \l__enumext_item_text_vii_box 3546, 3587, 3594 \l__enumext_item_text_viii_box 3899, 3938, 3945 $\label{local_local_local_local_local} $$ l_enumext_item_text_X_box $$\underline{154}$$ \l__enumext_item_width_vii_dim ... 3274, 3319, 3327, 3328 \l__enumext_item_width_viii_dim . . 3632, 3677, 3685, 3686 \l__enumext_item_width_X_dim 154 \l__enumext_itemindent_X_dim 59 \l__enumext_itemsep_vii_skip 3616 \l__enumext_itemsep_viii_skip 3967 \l__enumext_joined_item_aux_vii_int .. 3313, 3314, 3315, 3316, 3322
- \l__enumext_joined_item_aux_viii_int . 3671, 3672, 3673, 3674, 3680 \l__enumext_joined_item_aux_X_int 154 __enumext_joined_item_vii:w 95, 96, 3469, 3470, 3472, 3472 \l__enumext_joined_item_vii_int .. 3284, 3285, 3288, 3290, 3296, 3301, 3306, 3311, 3313, 3319 __enumext_joined_item_viii:w . 102, 3802, 3803, 3805, 3805 $\l_{\text{enumext_joined_item_viii_int}}$. 3642, 3643, 3646, 3648, 3654, 3659, 3664, 3669, 3671, 3677 \l__enumext_joined_item_X_int 154 \l__enumext_joined_width_vii_dim . 3317, 3324, 3327, 3577, 3589 \l__enumext_joined_width_viii_dim 3675, 3682, 3685, 3921, 3940 \l__enumext_joined_width_X_dim 154 __enumext_keyans_addto_prop:n 72, 2390, 2390, 2698, 3170 $\ensuremath{\mbox{\mbox{-}enumext_keyans_addto_seq:n}}\ . \ 73, 2464, 2464,$ 2700, 3172 __enumext_keyans_addto_seq_link: 2464, 2485, 2487, 3861 __enumext_keyans_anspic_code:nnn . 89, 3161, 3164, 3164 __enumext_keyans_default_item:n . . 78, 2679, 2679, 2715 \l__enumext_keyans_env_bool <u>20</u>, 2914, 2927, 3056, 3141 __enumext_keyans_fake_item: . . 789, 805, 2812 \l__enumext_keyans_item_opt_tl . 103, 97, 2512, 2525, 2531, 3846 $l_enumext_keyans_level_h_int ... 20, 594, 621,$ 2442, 3760, 3761 2437, 3055, 3060, 3155 __enumext_keyans_make_label: 33, 79, 2745, 2745, __enumext_keyans_mini_addvspace: 48, 87, 1227, 1227, 3085 __enumext_keyans_mini_right_cmd:n 51, 1325, 1348, 1348 __enumext_keyans_mini_set_vskip: . 47, 1165, 1165, 1229 __enumext_keyans_multi_addvspace: 87, 1014, 1025, 3110 __enumext_keyans_multi_set_vskip: 44, 1014, 1014, 1027 __enumext_keyans_multicols_start: 87, 3089, 3091, 3091 __enumext_keyans_multicols_stop: . 88, 1352, 3116, 3116, 3140 __enumext_keyans_parse_keys:n 3034, 3069, 3069 $\label{locality} $$ l_enumext_keyans_pic_above_int . $$ \underline{126}, 3250, $$$ 3251, 3253 \l__enumext_keyans_pic_above_skip .. 91, 126, 3194, 3234 __enumext_keyans_pic_arg_two: 90, 3192, 3222, \l__enumext_keyans_pic_below_int . <u>126</u>, 3250, 3251, 3254 $l_enumext_keyans_pic_body_seq$. . 89-91, 126, 3159, 3199, 3258

__enumext_keyans_pic_do:n 91, 3199, 3201, 3242,

3242, 3246
\lenumext_keyans_pic_level_int <u>20</u> , 1315,
2199, 2393, 2432, 2467, 2545, 3210, 3211
\enumext_keyans_pic_row:n 91, 3244, 3247, 3247
\enumext_keyans_pic_safe_exec: 90, 3188,
<u>3208,</u> 3208
\enumext_keyans_pic_skip_abs:N 90, 3217,
3217, 3233
\lenumext_keyans_pic_width_dim . <u>126</u> , 3249,
3256
\enumext_keyans_redefine_item: 78, <u>2703</u> ,
2703, 2809
\enumext_keyans_ref: 36, <u>646</u> , 664, 2811
\enumext_keyans_ref:n 36, 643, 646, 646
\enumext_keyans_safe_exec: . 3033, 3049, 3049
\enumext_keyans_save_start_line: . 28, 266,
266, 3057, 3215, 3765
\enumext_keyans_show_ans: <u>2508</u> , 2516, 2535
\enumext_keyans_show_item_opt: . 2508, 2523,
2696, 3184, 3927
\enumext_keyans_show_left:n . 78, 2508, 2508,
2694, 3179
\enumext_keyans_show_pos: <u>2508</u> , 2520, 2543
\enumext_keyans_starred_item:n 78, 2691,
2691, 2711
\enumext_keyans_store_ref: 72, 2411, 2411,

2699, 3171, 3859
\enumext_keyans_store_ref_aux_i: 73, 2411,
2423, 2426
\enumext_keyans_store_ref_aux_ii: 73, 2411,
2452, 2454
\lenumext_keyans_tmpa_tl 24, 97, 2693, 2697
-
\enumext_keyans_wrapper_opt:n 1963, 2531
\lenumext_label_copy_i_tl 2323, 2430, 2435,
2440, 2445
\lenumext_label_copy_v_tl 2440
<pre>\lenumext_label_copy_v_tl 2440 \l_enumext_label_copy_vi_tl 2435</pre>
\lenumext_label_copy_v_tl
<pre>\lenumext_label_copy_v_tl 2440 \l_enumext_label_copy_vi_tl 2435</pre>
\lenumext_label_copy_v_tl
\lenumext_label_copy_v_tl
\lenumext_label_copy_v_tl
\\ \enumext_label_copy_v_tl
\\enumext_label_copy_v_tl
\\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\\enumext_label_copy_v_tl
\\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\\enumext_label_copy_v_tl
\\enumext_label_copy_v_tl
\\enumext_label_copy_v_tl
\\enumext_label_copy_v_tl
\\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\\enumext_label_copy_v_tl
\\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\\enumext_label_copy_v_tl
\\enumext_label_copy_vi_tl
\lenumext_label_copy_vi_tl
\\enumext_label_copy_vi_tl

```
\l__enumext_labelsep_i_dim . . . 2540, 2575, 3865,
\l__enumext_labelsep_v_dim . . . . . . . . . . 3100
\l__enumext_labelsep_vii_dim . 3269, 3278, 3320,
    3515, 3575, 3591
\l__enumext_labelsep_viii_dim 3627, 3636, 3678,
    3919, 3942
\l__enumext_labelwidth_i_dim . 2540, 2575, 3865,
\l__enumext_labelwidth_v_dim ..... 3100
\l__enumext_labelwidth_vii_dim ... 3269, 3277,
    3320, 3568, 3572, 3590
\l__enumext_labelwidth_viii_dim . . 3627, 3635,
    3678, 3912, 3916, 3941
\l__enumext_leftmargin_tmp_v_bool . 90, 3224
\l__enumext_leftmargin_tmp_X_bool .... 59
\l__enumext_leftmargin_tmp_X_dim ..... 59
\l__enumext_leftmargin_X_dim ..... 59
\__enumext_level: <u>187</u>, 187, 538, 541, 542, 551, 553,
    792, 796, 800, 867, 871, 875, 879, 963, 965, 967, 969,
    1002, 1004, 1006, 1008, 1012, 1052, 1055, 1074, 1083,
    1089, 1094, 1098, 1109, 1113, 1114, 1119, 1155, 1159,
    1332, 1338, 1385, 1387, 1389, 1392, 1399, 1401, 1403,
    1406, 2012, 2014, 2016, 2044, 2045, 2047, 2103, 2111,
    2115, 2119, 2361, 2364, 2365, 2635, 2636, 2640, 2641,
    2642, 2649, 2651, 2655, 2656, 2659, 2665, 2666, 2721,
    2724, 2726, 2733, 2734, 2735, 2738, 2741, 2871, 2873,
    2920, 2933, 2940, 2951, 2953, 2956, 2957, 2959, 2964,
    2971, 2974, 2976, 2978, 2979, 2980, 2981, 2984, 2990,
    2995, 3001, 3004, 3006, 3012
\l__enumext_level_h_int .. <u>20</u>, 219, 241, 254, 577,
    614, 1820, 1836, 2317, 2334, 3405, 3406
\l__enumext_level_int . 82, 20, 189, 228, 240, 255,
    975, 1126, 1319, 1814, 1842, 1922, 2294, 2304, 2310,
    2316, 2324, 2332, 2339, 2825, 2886, 2887, 2897, 2904,
    2918, 2931, 2986, 3064, 3151, 3443, 3453, 3768
\__enumext_list_arg_two_i: ..... 2791
\__enumext_list_arg_two_ii: . . . . . . . . . 2791
\__enumext_list_arg_two_iii: ..... 2791
\__enumext_list_arg_two_iv: ..... 2791
\__enumext_list_arg_two_v: . 78, 2791, 3039, 3225
\__enumext_list_arg_two_vii: .... 2831, 3387
\__enumext_list_arg_two_viii: .... 2831, 3743
\l__enumext_listoffset_v_dim ..... 3102
\l__enumext_listparindent_vii_dim .... 3578
\l__enumext_listparindent_viii_dim ... 3922
\__enumext_log_answer_vars: . 29, 317, 325, 1929
\__enumext_log_global_vars: . 29, 317, 317, 1928
\__enumext_make_label: 33, 76, 77, 79, 2729, 2729,
l_enumext_mark_answer_sym_tl . 67, \underline{121}, 1969,
    2173, 2376, 2547, 2560, 3869
\label{local_enumext_mark_position_str} \ \ 121, 1973, 1974,
    1997, 1998, 2171
\l__enumext_mark_ref_sym_tl . . <u>121</u>, 1983, 2274,
\__enumext_mini_addvspace: . . 47, 84, 1148, 1148,
\__enumext_mini_addvspace_vii: 49, 1301, 1301,
    3345
\__enumext_mini_addvspace_viii: 49, 1301, 1307,
__enumext_mini_env* .......... 1042
\__enumext_mini_right_cmd:n . 50, 51, 1327, 1329,
```

1220
1329
\enumext_mini_set_vskip: . 45, 1049, 1049, 1150
\enumext_mini_set_vskip_vii: 48 , $\underline{1244}$, 1244 ,
1303
\enumext_mini_set_vskip_viii: 48, 1244, 1266,
1309
\enumext_minipage:w 30, 338, 340, 1044, 3256,
3577, 3921
\lenumext_minipage_active_v_bool 87, 88,
3083, 3108, 3121, 3129
\g_enumext_minipage_active_vii_bool 93,
3356, 3361, 3373
\lenumext_minipage_active_vii_bool . 3341,
3352
\genumext_minipage_active_viii_bool 3714,
3719, 3731
\lenumext_minipage_active_viii_bool 3699,
3710
\genumext_minipage_active_X_bool 154
\lenumext_minipage_active_X_bool 73
\g_enumext_minipage_after_skip 73, 1248, 1260,
3371, 3729
\lenumext_minipage_after_skip 45, 46, 85, 88,
<u>73</u> , 1065, 1080, 1100, 1116, 1131, 1137, 1143, 1157,
1167, 1176, 1179, 1191, 1209, 1220, 1236, 1268, 1281,
1295, 3021, 3138
\genumext_minipage_center_vii_bool . 3365,
3374
\genumext_minipage_center_viii_bool 3723,
3732
\g_enumext_minipage_center_X_bool 154
\lenumext_minipage_hsep_v_dim 87, 3081
\l_enumext_minipage_hsep_vii_dim 3339
\lenumext_minipage_hsep_viii_dim 3697
\lenumext_minipage_left_skip $45, 87, 73, 1057,$
1072, 1091, 1106, 1153, 1163, 1168, 1174, 1183, 1200,
1212, 1232, 1242, 1246, 1251, 1255, 1269, 1273, 1287,
1305, 1311
\lenumext_minipage_left_v_dim 87,3079,3087
\lenumext_minipage_left_vii_dim 3335, 3347
\lenumext_minipage_left_viii_dim 3693, 3705
\lenumext_minipage_left_X_dim 73
\g_enumext_minipage_right_skip 73, 1247, 1252,
1256, 3364, 3722
\lenumext_minipage_right_skip . 45, 73, 1061,
1076, 1096, 1111, 1169, 1175, 1187, 1205, 1216, 1270,
1277, 1291, 1339, 1356
\lenumext_minipage_right_v_dim 87, 1350,
1355, 3077, 3081
\genumext_minipage_right_vii_dim 93, 3343,
3363, 3376
\lenumext_minipage_right_vii_dim 93,3333,
3338, 3344
\genumext_minipage_right_viii_dim 3701,
3721, 3734
\l_enumext_minipage_right_viii_dim 3691,
3696, 3702
\g_enumext_minipage_right_X_dim 154

\genumext_minipage_right_X_skip 154
\g_enumext_minipage_stat_int $.84, 87, 73, 1344,$
1361, 2960, 3014, 3019, 3084, 3131, 3136
\genumext_miniright_code_vii_tl . 93, 3369,
3375
\genumext_miniright_code_viii_tl 3727, 3733
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

```
\__enumext_multi_addvspace: . . . 44, 85, 997, 997,
\__enumext_multi_set_vskip: . . 43, 961, 961, 999
\l__enumext_multicols_above_ii_skip . . . 980
\l__enumext_multicols_above_iii_skip . . 986
\l__enumext_multicols_above_iv_skip . . . 992
\l__enumext_multicols_above_v_skip 1016, 1030,
\l__enumext_multicols_above_X_skip .... 67
\l__enumext_multicols_below_v_skip 1020, 1034,
\l__enumext_multicols_below_X_skip .... 67
\__enumext_multicols_start: 84, 2966, 2968, 2968
\__enumext_multicols_stop: 85, 1334, 2998, 2998,
\__enumext_newlabel:nn 26, 31, 71, 381, 381, 2350,
    2458
\l__enumext_newlabel_arg_one_tl 26, 31, 71, 73,
    143, 2273, 2343, 2351, 2447, 2459, 2497
\l__enumext_newlabel_arg_two_tl 26, 31, 70, 143,
    2297, 2307, 2321, 2337, 2352, 2434, 2439, 2444, 2460
\__enumext_parse_keys:n ... 54, 2867, 2892, 2892
\__enumext_parse_keys_vii:n 54, 3382, 3413, 3413
\__enumext_parse_keys_viii:n . 3739, 3773, 3773
\__enumext_parse_save_key:n 65, 2037, 2042, 2042
\__enumext_parse_save_key_vii:n 65, 2032, 2042,
\__enumext_parse_serie:n ........ 94
\__enumext_parse_series:n . . 54, 83, 1518, 1518,
    2900, 3419
\__enumext_parse_store_keys:n ..... 83
\l__enumext_parsep_i_skip 978, 980, 1129, 1177
\l__enumext_parsep_ii_skip . . . . 984, 986, 1135
\l__enumext_parsep_iii_skip . . . 990, 992, 1141
\l__enumext_parsep_vii_skip ..... 3579
\l__enumext_parsep_viii_skip ..... 3923
\l__enumext_partopsep_v_skip . 1032, 1036, 1203,
    1207, 1214, 1218, 1234, 1238
\l__enumext_partopsep_viii_skip ..... 1279
\__enumext_phantomsection: 31, 345, 374, 378, 394
\__enumext_print_footnote: . . . 2594, 2617, 3597,
\__enumext_print_keyans_box:NN 67, 2165, 2165,
    2178, 2363, 2539, 2574, 3865, 3880
\l__enumext_print_keyans_i_tl ... 4006, 4028
\l__enumext_print_keyans_ii_tl ... 4010, 4029
\l__enumext_print_keyans_iii_tl .. 4014, 4030
\l__enumext_print_keyans_iv_tl ... 4018, 4031
\l__enumext_print_keyans_starred_tl 106, 107,
    <u>111</u>, 4002, 4045
\l__enumext_print_keyans_vii_tl 106, 4022, 4032
\l__enumext_print_keyans_X_tl ..... <u>111</u>
\__enumext_printkeyans:nnn 106, 107, 4033, 4036,
    4036
\__enumext_redefine_item: . 77, <u>2668</u>, 2668, 2819
\l__enumext_ref_key_arg_tl 34, 37, 202, 531, 532,
    545, 576, 579, 590, 596, 607, 648, 649, 660
\label{local_enumext_ref_the_count_tl} \ \ 34, 37, 538, 541,
    544, 584, 586, 589, 601, 603, 606, 654, 656, 659
\__enumext_regex_counter_style: .. 27, 34, 197,
    197, 539, 585, 602, 655
\__enumext_register_counter_style:Nn . . 413,
    413, 418, 419, 420, 421, 422
```

__enumext_remove_extra_parsep_vii: .. 3397,

3606, 3606 $\verb|__enumext_remove_extra_parsep_viii: . 3753,$ 3957, 3957 __enumext_renew_footnote: ... 2594, 2598, 3549, \l__enumext_renew_the_count_v_tl 657, 666, 668 \l__enumext_renew_the_count_vii_tl 587, 616, \l__enumext_renew_the_count_viii_tl 604, 623, \l__enumext_renew_the_count_X_tl 37 __enumext_reset_global_bool: .. 293, 296, 305 __enumext_reset_global_int: . . . 293, 295, 299 __enumext_reset_global_tl: 293, 297, 311 __enumext_reset_global_vars: . 29, 62, 293, 293, 1936 \l__enumext_resume_active_bool *54*, *56*, 48, 1522, 1642 __enumext_resume_counter: . . 56, 57, 1640, 1646, 1653 __enumext_resume_counter:n . 54, 56, 1611, 1616, 1640, 1640, 1710, 1718 __enumext_resume_counter_save_ans: . . 56, 57, 1640, 1651, 1683 __enumext_resume_counter_series: 56, 57, 1640, 1649, 1666 \g__enumext_resume_int ... 48, 1563, 1657, 1658 __enumext_resume_last:n . . 54, 1518, 1524, 1537 \l__enumext_resume_name_tl 48, 1559, 1567, 1570, 1586, 1594, 1597, 1643, 1644, 1672, 1679 __enumext_resume_save_counter: 55, 1550, 1550, 3028, 3437 __enumext_resume_series:n . 56, 1482, 1607, 1607 __enumext_resume_starred: . 58, 1483, 1704, 1704 \g__enumext_resume_vii_int . . 95, 48, 1590, 1662, 1663 __enumext_safe_exec: 82, 2866, 2883, 2883 __enumext_safe_exec_vii: ... 3381, 3402, 3402 __enumext_safe_exec_viii: . . . 3738, 3758, 3758 \l__enumext_series_name_tl 56 \l__enumext_series_str . 55, 83, 1480, 1520, 1528, 1529, 1531, 1533, 1554, 1557, 1561, 1581, 1584, 1588, 2896, 3417 __enumext_set_parse:n 4099, 4116, 4116 \l__enumext_setkey_tmpa_int ... <u>106</u>, 4092, 4096 \l__enumext_setkey_tmpa_seq . . <u>106</u>, 4090, 4100, 4106, 4108, 4110, 4123 \l__enumext_setkey_tmpa_tl <u>106</u>, 4098, 4102 $\verb|\lower| \verb|\lower| l_{-}enumext_setkey_tmpb_seq . . \underline{106}, 4091, 4094,$ 4098, 4099 \l__enumext_setkey_tmpb_tl <u>106</u>, 4118, 4120, 4121 $\label{local_loc$ 2370, 2514, 2528, 3175, 3863 __enumext_show_length:nnn . . 40, 205, 205, 4192, 4193, 4194, 4195, 4196, 4197, 4198, 4199, 4200, 4201, 4207, 4208, 4209, 4210, 4211, 4212, 4213, 4214, 4215, 4216 \l__enumext_show_position_bool <u>121</u>, 1980, 2004, 2374, 2518, 2529, 3176, 3867 \g__enumext_standar_bool 27, 82, <u>20</u>, 218, 221, 239, 308, 1552, 1617, 1629, 1655, 1668, 1706, 1841, 1854 \l__enumext_standar_bool . 82, 85, <u>20</u>, 2302, 2315, 2331, 2889, 3027

```
\l__enumext_standar_first_bool 28, 82, 20, 244,
    1539, 1686, 1748, 1755
\__enumext_standar_item_vii:w . 96, 3477, 3479,
\__enumext_standar_item_viii:w 102, 3810, 3812,
    3812
\__enumext_standar_ref: ... 34, 529, 549, 2821
\__enumext_standar_ref:n ... 34,521,529,529
\g__enumext_standar_series_tl . 48, 1541, 1542,
    1708, 1711
\g__enumext_starred_bool 27, 94, 95, 20, 227, 230,
    253, 309, 1579, 1622, 1633, 1660, 1675, 1714, 1819,
    1860,\, 2293,\, 2303,\, 2333,\, 2428,\, 2915,\, 2928,\, 3377
\l__enumext_starred_bool . 94, 95, <u>20</u>, 2226, 2234,
    2318, 2359, 3410, 3436
\__enumext_starred_columns_set_vii: . . 3263,
    3263, 3390
\__enumext_starred_columns_set_viii: . 3621,
    3621, 3746
l_enumext_starred_first_bool ... 28, 20, 258,
    1544, 1695, 1748, 1755
\__enumext_starred_item:nn . . . 2645, 2645, 2674
\__enumext_starred_item_exec: . 103, 3855, 3855,
\__enumext_starred_item_vii:w . 96, 3476, 3495,
    3495
\__enumext_starred_item_vii_aux_i:w . . 3495,
    3500, 3503
\__enumext_starred_item_vii_aux_ii:w . 3495,
    3501, 3506, 3508
\__enumext_starred_item_vii_aux_iii:w 3495,
    3511, 3520
\__enumext_starred_item_viii:w 102, 103, 3809,
    3828, 3828
\__enumext_starred_item_viii_aux_i:w . . 103,
    <u>3828</u>, 3833, 3836
\__enumext_starred_item_viii_aux_ii:w . 103,
    <u>3828</u>, 3834, 3848, 3850
\__enumext_starred_joined_item_vii:n . 92, 96,
    3282, 3282, 3474
\__enumext_starred_joined_item_viii:n .. 99,
    102, 3640, 3640, 3807
\__enumext_starred_ref: .... 36, 574, 612, 2849
\__enumext_starred_ref:n .... 35, 568, 574, 574
\g__enumext_starred_series_tl . 48, 1546, 1547,
    1716, 1719
\__enumext_start_from:NNn 37, 671, 671, 684, 706
\l__enumext_start_i_int .... 1658, 1670, 1689
\__enumext_start_item_tmp_vii: 93, 3393, 3459,
\__enumext_start_item_tmp_viii: .. 100, 3749,
    3792, 3792
\__enumext_start_item_vii:w . 96, 97, 3487, 3492,
    3517, 3524, 3526, 3526
\__enumext_start_item_viii:w . . 102, 3820, 3825,
    3853, 3883, 3883
\g__enumext_start_line_tl 28, 131, 246, 260, 314,
    1884, 1889, 1894, 1908, 1913, 1918
\__enumext_start_list:nn 30, 79, 90, 332, 334, 2870,
    3036, 3189, 3385, 3741
\__enumext_start_mini_vii: . 94, <u>3331</u>, 3331, 3428
\__enumext_start_mini_viii: ... 101, 3689, 3689,
    3784
\__enumext_start_save_ans_msg: 58, 1732, 1732,
```

1757
\enumext_start_store_level: . 83, 2869, 2909,
2909
\enumext_start_store_level_vii: . 95, 3384,
3439, 3439
\lenumext_start_vii_int 1663, 1677, 1698
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\enumext_stop_item_tmp_vii: . 93, 95, 97, 3392,
3396, 3461, 3528
\enumext_stop_item_tmp_viii: 100, 102, 3748,
3752, 3794, 3885
\enumext_stop_item_vii: 97, 98, 3528, <u>3582</u> , 3582
\enumext_stop_item_viii: 104, 3885, <u>3933</u> , 3933
\enumext_stop_list: 30, <u>332</u> , 335, 2879, 3046,
3202, 3398, 3755
\enumext_stop_mini_vii: 93, 95, 3350, 3350, 3432
\enumext_stop_mini_viii: 101, 3689, 3708, 3788
\enumext_stop_save_ans_msg: . 58, 1732, 1737,
1926
\enumext_stop_store_level: 83, 2880, 2909,
2938
\enumext_stop_store_level_vii: 95, 3399,
3439, 3449
\lenumext_store_active_bool 24, 59, 83, 94, 97,
1687, 1696, 1764, 2191, 2913, 2926, 3051, 3059, 3147,
3206, 3441, 3451, 3767
\enumext_store_active_keys:n 64, 65, 2010,
2010, 2906
\enumext_store_active_keys_vii:n 64,65,94,
<u>2010</u> , 2020, 3420
\enumext_store_addto_prop:n 66, 72, 2085, 2085,
2093, 2213, 2409, 3858
\enumext_store_addto_seq:n 66 , 74 , 2094 , 2094 ,
2098, 2105, 2119, 2127, 2136, 2154, 2162, 2277, 2502
\lenumext_store_anskey_arg_tl 24,69,97,
2219, 2228, 2230, 2236, 2244, 2247, 2257, 2262, 2265,
2271, 2277
\enumext_store_anskey_code:nnnn . 68, 2208,
<u>2211</u> , 2211
\enumext_store_anskey_show_left:n 71, 2218,
<u>2368,</u> 2368
\enumext_store_anskey_show_wrap:n 71 , 2356 ,
2356, 2372, 2387
\lenumext_store_columns_break_bool . 2185,
2225
2225 \lenumext_store_columns_join_int 97, 2233,
2225 $ \label{eq:columns_join_int} $$ 1_enumext_store_columns_join_int $$ $\underline{97}, 2233, $$ $2238 $$
2225 \lenumext_store_columns_join_int 97, 2233,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279
2225 \lenumext_store_columns_join_int 97, 2233,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 _enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259
2225 \lenumext_store_columns_join_int 97, 2233, 2238 _enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep_tl 1966, 2403, 2405, 2476, 2480, 3841, 3843
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97 \lenumext_store_keyans_label_tl 24, 72-74,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep_tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97 \lenumext_store_keyans_label_tl 24, 72-74,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep_tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97 \l_enumext_store_keyans_label_tl 24, 72-74, 103, 97, 2392, 2395, 2398, 2405, 2407, 2409, 2466, 2469, 2472, 2478, 2483, 2493, 2502, 3838, 3843, 3844, 3857, 3858, 3860
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep_tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97 \lenumext_store_keyans_label_tl 24, 72-74, 103, 97, 2392, 2395, 2398, 2405, 2407, 2409, 2466, 2469, 2472, 2478, 2483, 2493, 2502, 3838, 3843, 3844,
2225 \lenumext_store_columns_join_int 97, 2233, 2238 \enumext_store_internal_ref: 68, 70, 2216, 2279, 2279 \lenumext_store_item_symbol_sep_dim 2183, 2254, 2259 \lenumext_store_item_symbol_tl . 2181, 2245, 2249 \lenumext_store_keyans_item_opt_sep_tl 1966, 2403, 2405, 2476, 2480, 3841, 3843 \lenumext_store_keyans_item_opt_tl 97 \l_enumext_store_keyans_label_tl 24, 72-74, 103, 97, 2392, 2395, 2398, 2405, 2407, 2409, 2466, 2469, 2472, 2478, 2483, 2493, 2502, 3838, 3843, 3844, 3857, 3858, 3860

3455

```
\__enumext_store_level_open: . . 66, 2099, 2099,
\__enumext_store_level_open_vii: . 2130, 2130,
    3445
\g__enumext_store_name_tl . 24, 85, 97, 313, 320,
    321, 322, 323, 1740, 1766, 1883, 1888, 1893, 1907,
    1912, 1917, 1924
\l__enumext_store_name_tl . 24, 58, 60, 97, 1573,
    1576, 1600, 1603, 1691, 1700, 1735, 1744, 1745, 1766,
    1767, 1769, 1770, 1772, 1774, 1775, 1777, 1779, 1780,
    1804, 2087, 2089, 2096, 2345, 2346, 2382, 2449, 2450,
    2553, 2566, 3875
\l__enumext_store_ref_key_bool 68, 1986, 2214,
    2268, 2413, 2490
\l__enumext_store_save_key_vii_bool . . 2022,
\l__enumext_store_save_key_vii_tl 2024, 2025,
    2053, 2054, 2134, 2144, 2150, 2154
\l__enumext_store_save_key_X_bool .... 64
\l__enumext_store_save_key_X_tl . . 64, 65, 111
\l__enumext_store_upper_level_X_bool . . <u>111</u>
\l__enumext_store_write_aux_file_tl 26, 71, 73,
    143, 2348, 2354, 2456, 2462
\__enumext_storing_exec: 58, 59, 1742, 1758, 1762
\__enumext_storing_set:n . . 58, 1727, 1742, 1742
\l__enumext_the_counter_v_tl ..... 656
\l__enumext_the_counter_vii_tl ..... 586
\l__enumext_the_counter_viii_tl ..... 603
\l__enumext_the_counter_X_tl ..... 37
\__enumext_tmp:n 32, 36, 41, 47, 59, 66, 67, 72, 79, 84,
    85, 96, 112, 120, 146, 150, 154, 173, 784, 788, 1476,
    1487, 1723, 1731, 1783, 1801, 1956, 1991, 1992, 2009,
    2028, 2041, 2281, 2288, 2289, 2310, 2324, 2327, 2339,
    2415, 2422, 2791, 2830, 2831, 2863
\__enumext_tmp:nn 445, 466, 467, 495, 496, 508, 701,
    720, 765, 783, 841, 849, 850, 864, 929, 945, 946, 960,
    1365, 1381, 2578, 2593
\__enumext_tmp:nnn 509, 525, 526, 527, 528, 556, 572,
\__enumext_tmp:nnnnnn 721, 746, 749, 752, 754, 756,
    759, 762
\__enumext_tmp:w ..... 3982, 3985
\l__enumext_tmpa_vii_int ..... 3273, 3276
\l__enumext_tmpa_viii_int ..... 3631, 3634
\l__enumext_tmpa_X_int ..... 154
\l__enumext_topsep_v_skip 1018, 1022, 1172, 1185,
    1193, 1198, 1218, 1222, 3205, 3237
\l__enumext_topsep_vii_skip . . 1249, 1258, 1262
\l__enumext_topsep_viii_skip . 1271, 1293, 1297
\l__enumext_vspace_a_star_v_bool .... 1414
\l__enumext_vspace_a_star_vii_bool ... 1436
\l__enumext_vspace_a_star_viii_bool . . . 1447
\l__enumext_vspace_a_star_X_bool ..... 85
\__enumext_vspace_above: . . 51, 1382, 1382, 2947
\__enumext_vspace_above_v: . 52, 1410, 1410, 3075
\l__enumext_vspace_above_v_skip . . 1412, 1416,
    1418
\__enumext_vspace_above_vii: . . 52, 1432, 1432,
\l__enumext_vspace_above_vii_skip 1434, 1438,
\__enumext_vspace_above_viii: . 52, 1432, 1443,
\l__enumext_vspace_above_viii_skip 1445, 1449,
    1451
```

\lenumext_vspace_b_star_v_bool 1425	exp commands:
\lenumext_vspace_b_star_vii_bool 1458	\exp_after:wN 3985
\lenumext_vspace_b_star_viii_bool 1469	\exp_args:Ne 2903, 3973
\lenumext_vspace_b_star_X_bool 85	\exp_not:N . 45, 436, 544, 589, 606, 659, 798, 812, 813,
	824, 825, 836, 837, 2273, 2379, 2380, 2495, 2550, 2551,
\enumext_vspace_below_v: . 52, 1421, 1421, 3143	2563, 2564, 3872, 3873, 3982
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\	\exp_not:n 248, 262, 274, 281, 288, 544, 545, 589, 590,
	606, 607, 659, 660, 799, 1504, 1516, 2071, 2083, 2238,
1429	2249, 2259, 2273, 2274, 2351, 2459, 2497, 2499
\enumext_vspace_below_vii: 53, <u>1454</u> , 1454,	177 377 737 717 33 7 1377 1777 1777
3435	F
\lenumext_vspace_below_vii_skip 1456, 1460,	\fbox 1961
1462	file commands:
\enumext_vspace_below_viii: $. 53, \underline{1454}, 1465,$	\file_input_stop:
3790	first
\lenumext_vspace_below_viii_skip 1467, 1471,	font
1473	\footnote
\enumext_widest_from:nNNn 37, <u>685</u> , 685, 700,	•
712	\footnote
\genumext_widest_label_tl	\footnotemark
441	\footnotesize 2380, 2551, 2564, 3873
\lenumext_wrap_label_opt_v_bool 2687	\footnotetext 2596
\lenumext_wrap_label_opt_vii_bool 96,3486	
<pre>\lenumext_wrap_label_opt_viii_bool 102,</pre>	G
3819	\getkeyans 14, 105, <u>3971</u>
\lenumext_wrap_label_opt_X_bool 85	group commands:
\lenumext_wrap_label_v_bool 2683, 2687, 2695,	\group_begin: 2203, 2378, 2549, 2562, 3545, 3564,
2751	3871, 3898, 3908, 3993, 4027
\lenumext_wrap_label_vii_bool 96,3485,3490,	\group_end: 2209, 2385, 2556, 2569, 3574, 3586, 3878,
3498, 3566	3918, 3937, 3995, 4034
\lenumext_wrap_label_viii_bool . 102, 3818,	Н
3823, 3831, 3910	\hbadness 3593, 3944
\lenumext_wrap_label_X_bool <u>85</u>	hbox commands:
\enumext_wrapper_label_v:n 2753, 3184	\hbox_set:Nn 425
\enumext_wrapper_label_vii:n 3569	\hfill 475, 479, 484, 485, 1336, 1354, 2273, 2495, 3355, 3713
\enumext_wrapper_label_viii:n 3913	hook commands:
\enumext_zero_parsep: 46 , 1069 , 1124 , 1124	\hook_gput_code:nnn 9, 181, 185, 343
enumext* 5, <u>3379</u>	\hook_gset_rule:nnnn 344
enumXi 405	\hspace 3604, 3955
enumXii 405	\hyperlink 69, 74
enumXiii 405	\hyperlink 2273, 2495
enumXiv	\hypertarget
 -	\hypertarget
enumXv	
enumXvi $\underline{405}$	I
enumXvii $\underline{405}$	\IfHyperBoolean 351
enumXviii <u>405</u>	\IfPackageLoadedTF 11, 347, 361
Environments provide by enumext:	\ignorespaces 801
enumext* 22, 23, 25-27, 32, 35, 36, 39-42, 48, 49, 52-56,	\inputlineno
58, 60-70, 73, 75, 76, 81-83, 94, 95, 97, 98, 100, 102,	int commands:
104, 106, 107, 110, 112	\int_add:Nn 3315, 3673
enumext 22, 23, 25, 27, 32-38, 40-48, 50-56, 58, 60-70, 73,	\int_case:nn 975, 1126, 1814, 1836, 1874, 1898
75-77, 79, 80, 82, 83, 85, 86, 90, 91, 93, 95, 106, 107,	\int_compare:nNnTF . 577, 594, 614, 621, 1051, 1170,
109, 111	1315, 1319, 1323, 1922, 1941, 1947, 2195, 2199, 2393,
keyans* 22–28, 32, 35, 36, 39–42, 48, 49, 52, 53, 59, 60, 63,	2432, 2437, 2442, 2467, 2545, 2887, 2897, 2918, 2931,
64, 66, 72, 76, 81, 101, 110, 112	2970, 2986, 3000, 3014, 3060, 3064, 3093, 3118, 3131,
keyanspic 22–25, 28, 32, 33, 36, 50, 59, 60, 63, 66, 72–74,	3151, 3155, 3211, 3285, 3295, 3311, 3406, 3443, 3453,
88-91, 111	
keyans 22–25, 27, 28, 32, 33, 36–38, 40–42, 44, 47, 48,	3599, 3608, 3643, 3653, 3669, 3761, 3768, 3950, 3959,
50-52, 59, 60, 63, 64, 66, 72-74, 78-80, 86, 88-90, 93,	4096
50-52, 59, 00, 03, 04, 00, /2-/4, /o-ou, 00, 00-90, 93, 102, 110, 111	\int_compare_p:nNn 219, 228, 240, 241, 254, 255,
Environments:	1820, 1842, 2294, 2304, 2316, 2317, 2332, 2334
	\int_decr:N 3314, 3672
list	\int_eval:n . 330, 2089, 2346, 2380, 2450, 2551, 2564,
lrbox	2806, 2848, 3303, 3661, 3873
minipage 26, 30, 42, 44, 45, 88–91, 97, 98, 104	\int_from_alph:n 679, 693
multicols	\int_from_roman:n 681, 695

Variable and de Min	
\int_gadd:Nn	item*-sep
\int_gdecr:N 1823, 1827, 1830, 1833, 1845 \int_gincr:N 1657, 1662, 2206, 2505, 2633, 2663, 2701,	itemindent
2960, 3084, 3173, 3463, 3541, 3796, 3862	keys 94
\int_gset:Nn 1867, 2610	labelsep 33, 77, 80, 97
\int_gset_eq:NN 1556, 1563, 1569, 1575, 1583, 1590,	labelwidth
1596, 1602, 2607	label 23, 32, 34, 37, 91
\int_gzero:N . 301, 302, 303, 1344, 1361, 1953, 3019,	lisparindent
3136, 3617, 3968	list-indent 23, 38, 39, 90
\int_if_exist:NTF 1531, 1567, 1573, 1594, 1600, 1777	list-offset 38, 39
\int_incr:N 2886, 3055, 3210, 3405, 3462, 3760, 3795	listparindent
\int_mod:nn 3610, 3961	mark-ans 25, 63, 66, 71
\int_new:N . 20, 21, 22, 23, 24, 48, 49, 73, 89, 101, 108,	mark-pos
128, 129, 137, 138, 139, 140, 151, 157, 158, 159, 160,	mark-ref
161, 1533, 1780	mini-env 24, 42, 45, 50, 51, 66, 76, 84, 87, 93, 94, 100, 101
\int_set:Nn 675, 679, 681, 1670, 1677, 1689, 1698, 2233,	mini-sep
3250, 3251, 3273, 3284, 3290, 3306, 3593, 3631, 3642,	miniright*
3648, 3664, 3944, 4092	miniright
\int_set_eq:NN 1658, 1663, 3313, 3671	minirigth*
\int_sign:n	
\int_step_function:nnN 2310, 2324, 2339	minirigth
	no-store
\int_step_inline:nnn 3252	noitemsep 38, 46
\int_to_roman:n 189, 2290, 2328	nosep
\int_use:N 323, 328, 329, 1052, 1672, 1679, 1691, 1700,	parindent
2806, 2825, 2848, 2904, 2971, 2980, 2995, 3001, 3288,	parsep
3289, 3301, 3646, 3647, 3659	partopsep 38
\int_zero:N 3602, 3953	ref 23, 27, 34-36, 110
\c_one_int . 3273, 3292, 3298, 3304, 3308, 3311, 3631,	resume* 23, 53, 54, 58, 59, 65, 85
3650, 3656, 3662, 3666, 3669	resume
\c_zero_int 2294, 2304, 2316, 2317, 2332, 2334, 3443,	rightmargin 38
3453, 3613, 3964	save-ans 24, 29, 54–58, 60–62, 64–66, 68, 72, 73, 78, 83
vitem 30, 41, 42, 67, 76, 88, 89, 91, 93, 100	86, 89, 95, 102, 103, 105, 106, 110
vitem 76, 78, 95, 97, 102, 104, 336, 2107, 2113, 2138, 2146, 2230,	save-key
2469, 2472, 2670, 2705, 3391, 3393, 3747, 3749, 3860	
item*	save-pos
item-pos*	save-ref 26, 31, 63, 66, 68–70, 72, 74, 78, 103
	save-sep
item-sym* <u>2578</u>	series
itemindent 23, 80	show-ans
titemindent 80	show-length
itemindent 765	show-pos 25, 63, 64, 67, 68, 71, 74, 78, 103
itemsep 89, 90	start 24, 27, 37, 54, 66
itemsep 3226, 3232	store-brk
itemwidth 3280, 3324, 3328, 3638, 3682, 3686	store-key 64
	topsep 38
K	widest
keyans	wrap-ans
keyans*	wrap-label*
<u></u>	
keyanspic	wrap-label
Keys for environments provide by enumext:	wrap-opt 63, 66
above*	keys commands:
above	\keys_define:nn 447, 469, 498, 511, 558, 629, 703, 723
after 40-42, 85, 88, 95, 101	767, 786, 843, 852, 931, 948, 1367, 1478, 1725, 1785,
align 24, 33, 79, 97	1958, 1994, 2030, 2035, 2179, 2580, 3998, 4061
before* 40, 41, 84, 94, 101	\l_keys_key_str4177
before	\keys_precompile:nnN . 106, 3997, 4000, 4004, 4008
below*	4012, 4016, 4020
below	\keys_set:nn . 461, 955, 1372, 1377, 1619, 1624, 1711
check-ans 24, 25, 27, 28, 58–63, 66, 68, 74, 76, 77, 84, 85,	1719, 2222, 2899, 2903, 3071, 3418, 3777, 4063, 4064,
98, 110	4065, 4066, 4067, 4068, 4069, 4070, 4071, 4072, 4073,
	4005, 4000, 4007, 4000, 4009, 4070, 4071, 4072, 4075, 4013
columns-sep	40/4, 40/5, 4113 keyval commands:
columns	•
first	\keyval_parse:NNn 1492, 2060
font	•
item-pos* 68, 69, 75	L
:+	1-4-1

Labels provide by enumext:	\msg_new:nnn 4130, 4134, 4138, 4142, 4147, 4151, 4155,
\Alph* 32	4159, 4165, 4171, 4175, 4179, 4184, 4189, 4204, 4219,
\Roman* 32	4223, 4227, 4231, 4235, 4239, 4243, 4250, 4256, 4262,
\alph* 32	4266, 4270, 4275, 4280, 4285, 4290, 4294, 4299, 4304,
\arabic* 27, 32	4308, 4313, 4317, 4322, 4327, 4332, 4336, 4340
\roman* 32	\msg_term:nnnn . 1734, 1739, 2815, 2825, 2854, 2859
\labelsep 90	\msg_term:nnnnn 1888
\labelsep 3227, 3230	\msg_warning:nn 3016, 3133
labelsep <u>445</u>	\msg_warning:nnnn 1944, 1950, 2763, 2768, 3287, 3300,
\labelwidth 32, 90	3645, 3658
\labelwidth 3227, 3228	\msg_warning:nnnnn 1883, 1893
labelwidth <u>445</u>	\multicolsep
\leftmargin 23, 80	\multicolsep 2985, 3106
\leftmargin 80, 3227	••
legacy commands:	N
\legacy_if:nTF 3529, 3532, 3886, 3889	\NeedsTeXFormat 3
\legacy_if_gset_false:n 1045	\newcounter 402
\legacy_if_set_false:n 3531, 3888	\NewDocumentCommand 1313, 2189, 3145, 3971, 4025, 4083
\legacy_if_set_true:n 3491, 3516, 3523, 3536, 3824,	\NewDocumentEnvironment . 2864, 3031, 3186, 3379, 3736
3852, 3893	\newlabel 31
\linewidth 84, 87	\newlabel 385
\linewidth 2955, 3081, 3249, 3276, 3337, 3634, 3695	no-store
\list	\noindent 93, 100
\list	\noindent . 2962, 3086, 3346, 3392, 3601, 3704, 3748, 3952
list-indent	\nointerlineskip 2962, 3086, 3346, 3704
list-offset	noitemsep
\listparindent 3229	\nopagebreak 1011, 1039, 1162, 1241, 1304, 1310
listparindent	\normalfont 2379, 2550, 2563, 3872
\lrbox 3546, 3899	nosep
M	p
\makebox 91	Packages:
\makebox 2169, 2171, 2725, 3560, 3568, 3572, 3912, 3916	•
\makelabel	enumext
	enumitem 31, 32
\makelabel 76, 79, 91	enumitem
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30
$\begin{tabular}{lllllllllllllllllllllllllllllllllll$	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929 \minipage 30	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929 \minipage 30 \minipage 340	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929 \minipage 30 \minipage 340 \miniright 10, 50, 1313, 3017, 3134	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929 \minipage 30 \minipage 340	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 mini-sep 929 \minipage 30 \minipage 340 \miniright 10, 50, 1313, 3017, 3134 \miniright* 10 mode commands:	enumitem
\makelabel 76,79,91 \makelabel 79,2731,2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 minipage 30 \minipage 340 \miniright 10,50, 1313, 3017, 3134 \miniright* 10 mode commands: \mode_if_vertical:TF 1000, 1028, 1151, 1230	enumitem
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 minipage 30 \minipage 340 \miniright 10, 50, 1313, 3017, 3134 \miniright* 10 mode commands: \mode_if_vertical:TF 1000, 1028, 1151, 1230 \mode_leave_vertical: 798, 812, 824, 836, 2138,	enumitem
\makelabel \ 76,79,91 \makelabel \ 79,2731,2747 \makesavenoteenv \ 367 mark-ans \ 1956 mark-pos \ 1956, 1992 mark-ref \ 1956 mini-env \ 929 mini-sep \ 929 \minipage \ 30 \minipage \ 340 \miniright \ 10,50, 1313, 3017, 3134 \miniright* \ 10 mode commands: \mode_if_vertical:TF \ 1000, 1028, 1151, 1230 \mode_leave_vertical: \ 798, 812, 824, 836, 2138, 2146, 2167, 2723, 3558	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923
\makelabel 76, 79, 91 \makelabel 79, 2731, 2747 \makesavenoteenv 367 mark-ans 1956 mark-pos 1956, 1992 mark-ref 1956 mini-env 929 minipage 30 \minipage 340 \miniright 10, 50, 1313, 3017, 3134 \miniright* 10 mode commands: \mode_if_vertical:TF 1000, 1028, 1151, 1230 \mode_leave_vertical: 798, 812, 824, 836, 2138,	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 90 \partopsep 90 \partopsep 90 \partopsep 2846, 3231
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 2846, 3231 partopsep 721
\makelabel \ 76,79,91 \makelabel \ 79,2731,2747 \makesavenoteenv \ 367 \mark-ans \ 1956 \mark-pos \ 1956, 1992 \mark-ref \ 1956 \mini-env \ 929 \minipage \ 30 \minipage \ 340 \miniright \ 10,50,1313,3017,3134 \miniright* \ 10 \mode commands: \mode_if_vertical:TF \ 1000,1028,1151,1230 \mode_leave_vertical: \ 798,812,824,836,2138, \ 2146,2167,2723,3558 \msg commands: \msg_error:nn \ 3062,3066,3153,3213,3408,3763, \ 3770,4076	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 90 \partopsep 90 \partopsep 90 \partopsep 90 \partopsep 2846, 3231 partopsep 9219 peek commands:
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 90 \partopsep 2846, 3231 partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815,
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 90 \partopsep 2846, 3231 partopsep 2846, 3231 partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832
\makelabel \ 76, 79, 91 \makelabel \ 79, 2731, 2747 \makesavenoteenv \ 367 mark-ans \ 1956 mark-pos \ 1956, 1992 mark-ref \ 1956 mini-env \ 929 minipage \ 30 \minipage \ 340 \miniright \ 10, 50, 1313, 3017, 3134 \miniright* \ 10 mode commands: \mode_if_vertical:TF \ 1000, 1028, 1151, 1230 \mode_leave_vertical: \ 798, 812, 824, 836, 2138, 2146, 2167, 2723, 3558 msg commands: \msg_error:nn \ 3062, 3066, 3153, 3213, 3408, 3763, 3770, 4076 \msg_error:nnn \ 534, 581, 598, 651, 1317, 1321, 1346, 1363, 1631, 1635, 1750, 3987, 3992, 4058, 4129 \msg_error:nnnn \ 2193, 2197, 2201, 3053, 3149, 3157	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 90 \partopsep 90 \partopsep 90 \partopsep 90 \partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832 \peek_meaning_remove:NTF 3475, 3808 \peek_remove_spaces:n 2709 \phantomsection 31
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 2846, 3231 partopsep 90 \partopsep 2846, 3231 partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832 \peek_meaning_remove:NTF 3475, 3808 \peek_remove_spaces:n 2709 \phantomsection 31 \phantomsection 374
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 90 \partopsep 90 \partopsep 90 \partopsep 90 \partopsep 90 \partopsep 12846, 3231 partopsep 90 \partopsep 12846, 3231 partopsep 12846, 3231 partopsep 1296 \partopsep 1296 \partopse
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \text{par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \text{parindent 3578, 3922} \text{parsep 2139, 2147, 2845, 3226, 3233, 3238} \text{partopsep 90} \text{partopsep 2846, 3231} \text{partopsep 2846, 3231} \text{partopsep 90} \text{partopsep 12846, 3482, 3499, 3510, 3801, 3815, 3832} \text{peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832} \text{peek_meaning_remove:NTF 3475, 3808} \text{peek_remove_spaces:n 2709} \text{phantomsection 374} \text{prg_commands:} \text{prg_do_nothing: 378}
\makelabel	enumitem
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 90 \partopsep 2846, 3231 partopsep 90 \partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832 \peek_meaning_remove:NTF 3475, 3808 \peek_remove_spaces:n 2709 \phantomsection 374 prg_commands: \prg_do_nothing: 378 \prg_new_protected_conditional:Npnn 191 \prg_replicate:nn 208
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \text{par . 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \text{parindent 3578, 3922} \text{parsep 43, 46, 89, 90} \text{parsep 2139, 2147, 2845, 3226, 3233, 3238} \text{parsep 2139, 2147, 2845, 3226, 3233, 3238} \text{parsep 246, 3231} \text{partopsep 90} \text{partopsep 90} \text{partopsep 90} \text{partopsep 12846, 3231} \text{partopsep 90} \text{partopsep 12846, 3482, 3499, 3510, 3801, 3815, 3832} \text{peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832} \text{peek_meaning_remove:NTF 3475, 3808} \text{peek_remove_spaces:n 2709} \text{phantomsection 31} \text{phantomsection 374} \text{prg_commands: 374} \text{prg_commands: 378} \text{prg_new_protected_conditional:Npnn 191} \text{prg_replicate:nn 208} \text{\text{prg_replicate:nn 208}} \text{\text{\text{prg_replicate:nn 208}} \text{\text{\text{prg_replicate:nn 208}} \text{\text{\text{\text{prg_replicate:nn 208}}} \text{\text{\text{\text{prg_replicate:nn 208}}}
\makelabel	enumitem 31, 32 expl3 91 footnotehyper 30 hyperref 25, 26, 30, 31, 69, 74, 97, 108 lua-visual-debug 45 multicol 22, 108 shortlst 91 \par 1011, 1039, 1162, 1241, 1304, 1310, 1339, 1356, 2358, 3006, 3021, 3123, 3138, 3261, 3364, 3371, 3601, 3615, 3722, 3729, 3952, 3966 \parindent 3578, 3922 \parsep 43, 46, 89, 90 \parsep 2139, 2147, 2845, 3226, 3233, 3238 parsep 721 \parskip 3579, 3923 \partopsep 90 \partopsep 90 \partopsep 2846, 3231 partopsep 90 \partopsep 721 peek commands: \peek_meaning:NTF 3468, 3482, 3499, 3510, 3801, 3815, 3832 \peek_meaning_remove:NTF 3475, 3808 \peek_remove_spaces:n 2709 \phantomsection 374 prg_commands: \prg_do_nothing: 378 \prg_new_protected_conditional:Npnn 191 \prg_replicate:nn 208

prop commands:	\skip_gzero_new:N 1247, 1248
\prop_count: N 321, 2089, 2346, 2382, 2450, 2553, 2566,	\skip_horizontal:N 813, 825, 837, 3561, 3575, 3919
3875	\skip_horizontal:n 799, 2168, 2176, 2724, 2726,
\prop_gput_if_not_in:Nnn 2087	3559, 3927
\prop_if_exist:NTF 1767, 3991	\skip_if_eq:nnTF . 978, 984, 990, 1054, 1088, 1129,
\prop_item:Nn 3994	1135, 1141, 1172, 1177, 1198, 1249, 1271, 1384, 1398,
\prop_new:N 1770	1412, 1423, 1434, 1445, 1456, 1467
\ProvidesExplPackage 4	\skip_new:N 69, 70, 74, 75, 76, 77, 78, 130, 171
n.	\skip_set:Nn . 963, 967, 1016, 1020, 1057, 1061, 1065,
R	1072, 1076, 1080, 1091, 1096, 1100, 1106, 1111, 1116,
\raggedcolumns 2994, 3112	1174, 1175, 1176, 1183, 1187, 1191, 1200, 1205, 1209,
\ref	1212, 1216, 1220, 1251, 1255, 1273, 1277, 1281, 1287,
ref <u>509, 556, 629</u>	1291, 1295, 3220, 3234
\refstepcounter 3538, 3895	\skip_set_eq:NN 2804, 2844, 2845, 3578, 3579, 3922,
regex commands:	3923
\regex_match:nnTF 193, 678, 680, 692, 694 \regex_replace_once:nnN 201	\skip_use:N 965, 969, 1004, 1008, 1012, 1032, 1036, 1055, 1074, 1083, 1089, 1094, 1098, 1109, 1113, 1114,
\renewcommand	
\RenewDocumentCommand 2602, 2670, 2705, 2731, 2747	1119, 1155, 1159, 1185, 1385, 1389, 1392, 1399, 1403, 1406, 3006
\RequirePackage	\skip_zero:N 2846, 2985, 3106, 3231, 3232
resume	\skip_zero_new:N 1167, 1168, 1169, 1246, 1268, 1269,
resume*	1270
rightmargin	\c_zero_skip . 978, 984, 990, 1055, 1089, 1129, 1135,
\Roman	1141, 1172, 1177, 1198, 1249, 1271, 1385, 1399, 1412,
\Roman	1423, 1434, 1445, 1456, 1467
\roman	\small 4003, 4007, 4011, 4015, 4019, 4023
\roman 422, 527, 4015	\star 2584
1 /3 //1 3	start 701
S	\stepcounter
save-ans $\dots \underline{1723}$	str commands:
save-key <u>2028</u>	\c_backslash_str 4149, 4153, 4157, 4161, 4162, 4163,
save-ref	4167, 4168, 4264, 4268, 4277, 4278, 4282, 4283, 4287,
save-sep	4288, 4319, 4320, 4324, 4329, 4330
scan commands:	\c_colon_str 2345, 2449, 3982
\scan_stop: 91, 3240, 3391, 3747, 3982, 3985	\c_left_brace_str 4229, 4233, 4246, 4252, 4258
seq commands:	\c_right_brace_str 4229, 4233, 4246, 4252, 4258
\seq_clear:N 4090	\str_case:nn 213, 268
\seq_const_from_clist:Nn 4078	\str_case:nnTF 1499, 1508, 2067, 2075
\seq_count:N 322, 3199, 4094	\str_clear:N 2896, 3417
\seq_gclear:N 2600, 2601	\str_count:n 208
\seq_gput_right: Nn 2096, 2613, 2614	\str_if_empty:NTF 1520, 1561, 1588
\seq_if_empty:NTF 2619, 4040, 4108	\str_if_eq:nnTF 2807, 2850
\seq_if_exist:NTF	\str_if_in:nnTF
\seq_map_function:NN	\str_new:N
\seq_map_inline:Nn 4047, 4052, 4087, 4109, 4110	\str_set:Nn 501, 502, 503, 1973, 1974, 1997, 1998 \string
\seq_map_pairwise_function: NNN 2621	\string
\seq_new:N 109, 110, 126, 152, 153, 1775	1118, 1131, 1137, 1143, 1174, 1175, 1176, 1179, 1189,
\seq_pop_left:NN	1193, 1202, 1209, 1214, 1222, 1251, 1252, 1255, 1262,
\seq_put_right:Nn 3159, 4106, 4123	1275, 1283, 1289, 1297, 3236
\seq_set_from_clist:Nn 4091	- 137 37 717 3 3 -
\seq_set_map_e:NNn 4100	T
\seq_show:N 4042	TeX and LaTeX 2_{ε} commands:
series <u>1476</u>	\@auxout 383
\setcounter 689, 693, 695, 2806, 2848, 3204	\@currenvir 213, 268
\setenumext 6, 107, $\underline{4083}$	\protected@write 383
\setlength 2140, 2148	text commands:
show-ans	\text_expand:n 3974
show-length <u>841</u>	\textasteriskcentered 1970, 1984
show-pos	\thepage 389
skip commands:	tl commands:
\skip_add:Nn . 980, 986, 992, 1002, 1006, 1030, 1034,	\c_space_tl 2531, 4191, 4206, 4229, 4233
1131, 1137, 1143, 1153, 1157, 1179, 1232, 1236, 3226	\tl_clear:N 474, 480, 1954, 2014, 2024, 2045, 2053,
\skip_eval:n 2139, 2147	2219, 2392, 2466, 3838
\skip_gset:Nn 1252, 1256, 1260	\tl_clear_new:N431

\tl_const:Nn	3869, 4118 \tl_set_eq:NN
\tl_if_blank:nTF	2749, 2750, 2756, 2871, 3037, 3183, 3369, 3565, 3576, 3580, 3727, 3909, 3920, 3926, 3930, 4028, 4029, 4030, 4031, 4032, 4045, 4102 token commands: \token_to_str:N
\tl_if_empty:nTF1609	\topsep 2140, 2148
\tl_if_exist:NTF	topsep
3839, 4085	U
\tl_map_inline:Nn	\u 202 use commands:
92, 98, 99, 100, 102, 103, 104, 105, 106, 107, 111, 114, 116, 117, 123, 124, 134, 135, 136, 143, 144, 145, 148, 165, 168 \tl_put_left:Nn 2111, 2144, 2228, 2537, 2572, 3857,	\use:N
3860 \tl_put_right:Nn 432, 542, 587, 604, 657, 2115, 2150, 2230, 2236, 2244, 2247, 2257, 2262, 2265, 2271, 2297, 2307, 2321, 2337, 2343, 2348, 2395, 2398, 2405, 2407, 2434, 2439, 2444, 2447, 2456, 2469, 2472, 2478, 2483, 2493, 3843, 3844	V \value 1557, 1563, 1570, 1576, 1584, 1590, 1597, 1603 \vspace 1046, 1389, 1392, 1403, 1406, 1416, 1418, 1427, 1429 1438, 1440, 1449, 1451, 1460, 1462, 1471, 1473, 2139, 2147, 3194, 3205, 3616, 3967
\tl_remove_all:Nn	W widest