

V1.0 2024-06-24*

©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 \(answers \) to these in memory using multicol and scontents packages and the laseq and laprop modules.

6 The storage system 10 1.1 Description and usage 6.1 Keys for storage system 11 1.2 The concept of left margin 3 6.1.1 Keys for label and ref 11 6.1.2 Keys for wrap and display . . . 11 1.3.1 Internal counters 3 1.3.2 Public dimension 3 6.1.3 Keys for debug and checking . . 12 1.3.3 Support for multicol 3 6.2 The command \anskey 12 1.3.4 Support for minipage 6.2.1 Keys for \anskey 12 1.3.5 The \label and \ref system . 4 6.3 The environment anskey* 13 1.3.6 Support for \footnote 4 The environments provided 4 6.3.1 Keys for anskey* 13 2.1 The environment enumext 6.4 The environment keyans 14 2.2 The environment enumext* 5 6.4.1 The \item* in keyans 14 5 2.3.1 Keys for $\int m^*$ 6.5 The environment keyanspic 15 5 2.4 The command \item in enumext* . . 6.5.1 The command \anspic 15 3 The command \setenumext 6.6 Printing stored content 16 The command \setenumextmeta 6 6.6.1 The command \getkeyans . . . 16 The keyval system 6 5.1 Keys for label and ref 6 6.6.2 The command \printkeyans . 16 5.2 Keys for spaces 7 Full examples 17 5.2.1 Vertical spaces 7 The way of non-enumerated lists 19 8 5.2.2 Horizontal spaces

Motivation and acknowledgments

5.6.1 The command \miniright . . . 10 5.6.2 The key mini-right 10

5.4 Keys for start, series and resume.

5.5 Keys for multicols

5.6 Keys for minipage

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:

9

9

10

11 Index of Documentation 23

12 Implementation 25

13 Index of Implementation 131

- 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 FTEX 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

Contents

^{*}This file describes a documentation for v1.0, last revised 2024-06-24.

[†]E-mail: «pablgonz@educarchile.cl».

§.1 Introduction enumext v1.0

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] and scontents[4] packages, need to have a modern TEX distribution such as TEX Live or MiKTEX. It has been tested with the standard classes provided by ETEX: 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
 - 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 + 3z3(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":

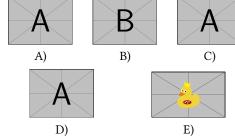
- 1. $(x-1)^2$
- 2. 3(x+y+z)
- 3. (a) False
 - (b) Very True!
- 4. (a) Yes

- (b) Yes, dnf
- (c) i. doesn't exist for now:(
- 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. $2\alpha + 2\delta = 90^{\circ}$
 - $\alpha = \delta$
 - 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 E) value
- B) value C) value

- - ii. very good
 - iii. obsolete
- 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:

```
      1. B), x = 5
      * 4. E), A duck

      2. D)
      * 5. D), "other note"

      3. C), some note
      *
```

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[6], 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[6] 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 of two main list environments enumext (vertical) and enumext* (horizontal), the environment anskey* and the command \anskey to "store content" and the environments keyans, keyans* and keyanspic for multiple choice. It also provides the commands \getkeyans to print individual stored content, \printkeyans to print all stored content, \miniright for minipage and \setenumext to config all $\lceil \langle key = val \rangle \rceil$ 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 fatal error and abort the load.

1.3.2 Public dimension

The package enumext only provides a single public dimension \itemwidth and is intended for user convenience only and is not for internal use as such. The dimension \itemwidth is rigid length and contains the "width of the content" of each \item regardless of labelwidth and labelsep.

If any package defines \itemwidth or they are user-defined \itemwidth in the document, the package will overwrite it without warning.

1.3.3 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. It can be used in all nesting levels of the environment enumext and the environment keyans and can together with the mini-env key. If you need to force a start a new column \columnbreak must be used (see §5.5).

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.

1.3.4 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 on "left side" and "right side" is always used with "aligned on top" [t]. It can be used in all nesting levels of the environment enumext and the environment keyans 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 §5.6).

1.3.5 The \label and \ref system

This package provides a user interface like the <code>enumitem[6]</code> package to customize the references which is activated by the <code>ref</code> key (§5.1), the standard <code>ETEX \label</code> and <code>\ref</code> commands work as usual. It also provides an "internal reference" system for the "stored content" by means of the key <code>save-ref</code> (§6.1.1) when the key <code>save-ans</code> (§6.1) is active.

The implementation of \label and \ref together with the save-ref key are compatible with the hyperref[8] package.

1.3.6 Support for \footnote

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

The best way to solve this is to use Jean-François Burnol footnotehyper[9] 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*}
```

2 The environments provided

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

```
enumext*
```

2.1 The environment enumext

The enumext is an environment that works in the same way as the standard enumerate environment provided by LTEX, \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 [$\langle key = val \rangle$] 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.
- i. This text is in the third level.
- ★ 2. This text is in the first level.

2.2 The environment enumext*

The enumext* 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 \item[$\langle custom \rangle$] work as usual. The options can be configured globally using \setenumext 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 or in the environment keyans*, but it can be nested within enumext and vice versa.
- Each "item" in the environment is placed within a minipage environment whose width is stored in the dimension \itemwidth that NOT includes labelwith, labelsep, only 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

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

2.3 The command \item*

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

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

The \item*, \item*[$\langle symbol \rangle$] and \item*[$\langle symbol \rangle$] [$\langle offset \rangle$] works like the numbered \item, but placing a $\langle symbol \rangle$ to the "left" of the $\langle label \rangle$ separated from it by the $\langle offset \rangle$ set by the 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 argument* '*' cannot be separated by spaces '__' 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).

©2024 by Pablo González L

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 and enumext*. The symbol can be in text or math mode, for example item-

```
item-pos* = \{\langle rigid\ length\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 columns \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
    (\the\itemwidth)
  \item The ninth
  \item[Z] The tenth (\the\itemwidth)
\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
- 7. The seventh X The eighth item is way too long for this and needs 8. The ninth two columns (187.14374pt)
- Z The tenth (84.76483pt)

The command \setenumext

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

🥑 The key save-ans that activate the "s*torage system*" must NOT be passed through this command and must be passed directly in the optional argument of the "first level" of the environment in which they are executed.

The command \setenumextmeta

```
\setenumextmeta \setenumextmeta \{\langle key \ name \rangle\} \{\langle key \ one = val, key \ two = val, ... \rangle\}
                              \stenumextmeta^* \{ \langle key \ name \rangle \} \{ \langle key \ one = val, \ key \ two = val, \dots \rangle \}
                              \setenumextmeta [\langle enumext^* \rangle] \{\langle key \ name \rangle\} \{\langle key \ one = val, \ key \ two = val, \dots \rangle\}
                              \setenumextmeta [\langle enumext, level \rangle] \{\langle key name \rangle\} \{\langle key-one = val, key-two = val, ... \rangle\}
```

The command \setenumextmeta adds a new "meta-key" for the environments enumext and enumext*, the $\{\langle key \ name \rangle\}$ must be different from those defined by the package. If the optional argument is not passed, the new "meta-key" will be created for the first level of the environment enumext.

The starred version * will create the new "meta-key" for the environment enumext* and for all levels of the environment enumext.

The keyval system 5

The $\langle key = val \rangle$ system used by the enumext package is implemented using l3keys 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.

Keys for label and ref

```
label = {\\alph* | \Alph* | \arabic* | \roman* | \Roman* \}
```

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*).

 $m{o}$ This key is intended to give the basic structure with which the $\langle label
angle$ 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 $\backslash emph\{\langle \ \ \ \ \ \ \ \ \ \ \ \}$ will return an error. For full customization of how $\langle label \rangle$ is displayed use the font or wrap-label keys.

```
ref = \{ \langle code \{ \lambda \} \rangle | \lambda \rangle
```

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: $ref=\{emph\{\langle alph^*\rangle\}\}$ 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\}
```

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 \}
```

Sets the labelwidth key pass the \(\int integer\) or converting the \(\string\) of the form \(\Alph\), \(\alphalph\), \(\Roman\) 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 *\label\range* 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 \}
```

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={\{\fbox\{\#1\}\}}\$ or you can create a command:

```
\NewDocumentCommand \labelbx { 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={\labelbx{#1}} or wrap-label={\labelbx*{#1}}.

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

default: empty

The same as the wrap-label key but also applies on $\idetit{\colored} \colored$.

5.2 Keys for spaces

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

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.

5.2.1 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 the keyans environment the default value is 2.0pt plus 1.0pt minus 1.0pt, and for the keyans* and enumext* environments it is available but *without* effect.

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 levels

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 \(\nu alue forbidde

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.

base-fix \(\text{value forbidden} \)

default: not used

This is a "meta-key" that does not receive an argument available only for the first level of environment enumext and environment enumext*. Fix the baseline when an environment enumext is nested in enumext* or vice versa and there is no material between the \item and the start of the environment for example \item \begin{enumext*} enumext*} within the environment enumext. Internally sets the keys topsep, above and above* at Opt.

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 \vspace* so that LTEX does not discard this space at page break.

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

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 *"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 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 *"not discardable"*.

5.2.2 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 <code>%pt</code>. Internally sets the value of <code>\rightmargin</code> for the current level.

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

default: Opt

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.

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

efault: 0*p*

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 is set in the environment enumext the $\langle label \rangle$ 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.

For the enumext* and keyans* environments the keys list-indent and list-offset have the same effect.

5.3 Keys for add code

The 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 $\{\text{list}(\langle arg\ one \rangle)\}\{\langle arg\ two\rangle\}$. Using the before* key does not allow access to the list parameters defined by $[\langle key=val\rangle]$.

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

default: 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 \begin{\list}{\langle} \langle arg one \rangle \{ \langle arg two} \{ \langle code \rangle \}\}.

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

default: not used

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\}\$ begin $\{\text{list}\}$ $\{\langle arg \ one \rangle\}$ $\{\langle arg \ two \rangle\}$.

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

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: \begin{\item} \large arg one \rangle \{\large arg two}\} \{\large code \rangle}\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.

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

default: not used

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

Keys for start, series and resume

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

default: 1

Sets the *start value* of the numbering on the current level. Internally $\langle string \rangle$ is converted and 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.

 $start* = {\langle integer expression \rangle}$

Sets the *start value* of the numbering on the current level. The $\{\langle integer\ expression \rangle\}$ must be passed between braces, internally is evaluated and pass to the counter defined by label key on the current level, i.e. it is equivalent to enter start={\dimeval{100*\value{chapter}} or start={100*\value{chapter}}.

of the following (keys) are "only" available for the enumext* environment and the "first level" of the enumext environment and are ignored if set when nested within 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.

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

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) 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 start or start* keys.

resume*

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 start or start* keys.

f For security reasons the series key will never save in {⟨series name⟩⟩ the keys series, resume, resume*, save-ans, save-key, start* 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 start value.

Keys for multicols 5.5

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.

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

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 $\lceil (number) \rceil$ inside the environment and $\lceil (number) \rceil \rceil \langle (text) \rangle$ outside the environment or via the after key.

5.6 Keys for minipage

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

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.

5.6.1 The command \miniright

```
\miniright \begin{enumext} [mini-env=\langle rigid length\rangle] \langle item's before\rangle \tem \miniright \langle content\rangle \tem \tem \miniff \mini
```

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 argument '*' inhibits the use of \centering command i.e. the usual LTEX justification is maintained in the minipage on the "right side".

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 \footnotemark[$\langle number \rangle$] inside the environment and \footnotetext[$\langle number \rangle$] { $\langle text \rangle$ } outside the environment or via the after key (see §1.3.6 for full support).

5.6.2 The key mini-right

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

```
mini-right = \{\langle content \rangle\}
```

default: not used

Set the *content* for the drawing or tabular to be placed in the minipage environment on the "right side" by starting it with \centering. The $\{\langle content \rangle\}$ must be passed between braces.

```
mini-right^* = \{\langle content \rangle\}
```

default: not used

Same as above, but without starting with \centering.

The keys mini-right and mini-right* has a *slightly different* implementation, the argument $\{\langle content \rangle\}$ is saved in a box and then printed outside the environment using *hooks*.

6 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 anskey*, keyans, keyans* and keyanspic are available.

```
\begin{enumext}[save-ans={\store name}\]
\item Text \anskey{answer}
\item Text
\begin{keyans}

\item Text \anskey{answer}
\item Text
\begin{keyanspic}

\limits \tem \text \anskey{answer}
\item \text \anskey{answer}
\item \text \anskey{answer}
\item \text \anskey{answer}
\item \text \anskey{answer}
\limits \text \anskey{answer}
\l
```

By executing the key save-ans={ $\langle store\ name \rangle$ } the entire structure of the environment (excluding the first level) including the optional arguments passed to the inner levels or the environment nested in it, along with the content passed to \anskey, the current $\langle labels \rangle$ for \item* and \anspic* in the environments keyans, keyans* and keyanspic will be stored in a $\langle sequence \rangle$ and at the same time will be stored (without the environment structure or optional arguments) in a $\langle prop\ list \rangle$.

The optional arguments of the inner levels or the nested environment are filtered by excluding all $\langle keys \rangle$ related to the "stored system" along with the keys series, resume and resume* when storing in $\langle sequence \rangle$.

6.1 Keys for storage system

The only $\langle keys \rangle$ available for all levels of the enumext environment and the enumext* environment are no-store and save-key, the rest of the $\langle keys \rangle$ described in this section must be passed directly in the optional argument of the "first level" of the environment in which the key save-ans is executed. The key save-ans should NOT be passed with the command \setenumext.

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

default: not set

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

```
save-key = \{\langle key \, list \rangle\}
```

default: not set

This key *overrides* the default "*stored keys*" of the optional arguments of the inner levels or nested environment that will be passed to the $\langle sequence \rangle$. The $\langle key \ list \rangle$ passed to this key ignores any $\langle keys \rangle$ in the "*stored system*" and must be passed between braces. For example, if we execute at a second level:

```
\begin{enumext}[save-ans={\store name\}]
\item Text \anskey{answer}
\item Text
\begin{enumext}[nosep, columns=2, save-key={columns=3}]
...
\end{enumext}
\end{enumext}
```

The $\langle keys \rangle$ that will be stored by default in the $\langle sequence \rangle$ would be nosep, columns=2, but using the key save-key={columns=3} will overwrite this and store it in the $\langle sequence \rangle$ only the key columns=3 ignoring all the others.

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

default: {, }

Sets the *text symbol* that will separate the current $\langle label \rangle$ to the *optional argument* passed to the \backslash item* and \backslash anspic* in the keyans, keyans* and keyanspic environments and storing them in the $\langle store\ name \rangle$ defined by the save-ans key. 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.

6.1.1 Keys for label and ref

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

default: false

Activates the "internal label and ref" mechanism for referencing "stored content" in $\langle store\ name \rangle$ set by save-ans key. To reference the location of the "stored content" within the environment you must use $\ref\{\langle store\ name:position \rangle\}$, where $\langle position \rangle$ corresponds to the position occupied by the "stored content" in the $\langle store\ name \rangle$ 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.

6.1.2 Keys for wrap and display

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

default: \fbox+\parbox{#1}

Wraps the *argument* passed to the \anskey and the *body* in anskey* environment referenced by $\{\#1\}$ when using the show-ans or show-pos keys. The $\{\langle code \rangle\}$ must be passed between braces and only affects the *argument* or *body* and NOT the "stored content" in the sequence and *prop list* $\{\langle store\ name \rangle\}$ set by save-ans key. If this key is passed using \setenumext it is necessary to use double ' $\{\#1\}$ '.

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

default: [{#1}]

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

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

default: false

Displays the *argument* passed to the \anskey, the *body* for anskey* environment, the $\langle label \rangle$ for \item* and \anspic* at the place where it is executed. If the optional argument is present in \item* or \anspic* it will be shown using wrap-opt key.

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

 $default: \ \ \ \ \ \ textasterisk centered$

Sets the *symbol* to be displayed in the left margin for \anskey, anskey*, \item* and \anspic* in the place where they are executed when using the key show-ans.

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

default: left

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.

6.1.3 Keys for debug and checking

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

default: false

Displays the *position* occupied by the "stored content" by \anskey, anskey*, \item* and \anspic* in the *prop list* $\{\langle store\ name \rangle\}$ 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 = $\{\langle true \mid false \rangle\}$

default: false

Enables the *checking answer* mechanism displaying an appropriate message on the terminal. This key works under the logic that each \item or \item* that does not open an inner level or nested environment contains "only one answer" or "only one execution" of the \anskey or anskey*. It is intended to be used in conjunction with the no-store key.

no-store (value forbidden)

default: not used

This is a meta-key that does not receive an argument and disables the structure stored in the sequence $\{\langle store\ name \rangle\}$ set by save-ans key at the entire level or a nested environment in which it runs. This key is intended for use in internal levels or nested enumext or enumext* environments in which you want to use enumext or enumext* but "without" using the \anskey, "without" use anskey*, "without" interfering with the check-ans key and "without" storing an unwanted structure in the sequence $\{\langle store\ name \rangle\}$.

6.2 The command \anskey

 $\anskey \anskey [\langle keys \rangle] \{\langle content \rangle\}$

The command \anskey takes a mandatory non empty argument $\{\langle content \rangle\}$ and "stores" it in the sequence and prop list $\{\langle store\ name \rangle\}$ set by save-ans key. By design the command cannot be nested or passed verbatim material in the argument and it is assumed that each numbered \item or \item* within the environment in which it is active it has a "single execution" of \anskey unless \item or \item* open a nested level or use the no-store key.

If save-ref key are active and the hyperlink and hyperlink and hyper

The \anskey command is available for all levels of the enumext environment and the enumext* environment, but is disabled for the keyans, keyans* and keyanspic environments.

6.2.1 Keys for \anskey

By default the $\{\langle content \rangle\}$ passed to \anskey when "storing" in the sequence $\{\langle store\ name \rangle\}$ has the form \item $\langle content \rangle$, the following $\langle keys \rangle$ allow modifying the way in which it is "stored" in the sequence.

```
break-col \( \sqrt{value forbidden} \) default: not used \( \text{Stores } \langle \content \rangle \right) \) in the sequence \( \langle \stores \ \name \rangle \right) \) of the form \( \columnbreak \rangle \text{item } \langle \content \rangle \).
```

 $item-join = \{\langle columns \rangle\}$ default: not set

Set the *number of columns* to be used for $\idetilde{\columns}$ and stores $\{\langle content \rangle\}$ in the *sequence* $\{\langle store\ name \rangle\}$ of the form $\idetilde{\columns} \rangle$ $\langle content \rangle$.

item-star $\langle value\ forbidden \rangle$ default: $not\ used$

Stores $\{\langle content \rangle\}$ in the sequence $\{\langle store\ name \rangle\}$ of the form $\backslash item^* \langle content \rangle$.

 $item-sym^* = \{\langle symbol \rangle\}$ default: $\$ \setminus star \$$

Sets the symbol for $\identermath{\mbox{`item*}}$ when using the key $\identermath{\mbox{`item*}}$ and stores $\{\langle content \rangle\}$ in the sequence $\{\langle storename \rangle\}$ of the form $\identermath{\mbox{`item*}} [\langle symbol \rangle] \langle content \rangle$. The symbol can be in text or math mode, for example $\identermath{\mbox{`item*}} [\langle storename \rangle]$ stores $\identermath{\mbox{`item*}} [\langle storename \rangle]$.

item-pos* = $\{\langle rigid\ length \rangle\}$ default: not set

Sets the *offset* for $\ideta = m^*$ when using the keys item-star and item-sym* and stores $\{\langle content \rangle\}$ in the sequence $\{\langle store\ name \rangle\}$ of the form $\ideta = m^* [\langle symbol \rangle] [\langle offset \rangle] \langle content \rangle$.

Example

```
\begin{enumext} [save-ans=test, show-ans=true]
\item* Text containing our instructions or questions. \anskey{\( \first \ answer \) \\
\item Text containing our instructions or questions.
\begin{enumext}
\item Question.\anskey{\( \second \ answer \) \\
\end{enumext}
\item Text containing our instructions or questions. \anskey{\( \seta \ third \ answer \) \\
\item Text containing our instructions or questions. \anskey{\( \seta \ torta \ answer \) \\
\end{enumext}
\end{en
```

- \star 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

6.3 The environment anskey*

 $anskey^* \setminus begin\{anskey^*\}[\langle key = val \rangle] \langle body content \rangle \setminus end\{anskey^*\}$

The environment anskey* takes a mandatory $\{\langle body\ content \rangle\}$ and "stores" it in the sequence and prop list $\{\langle store\ name \rangle\}$ set by save-ans key. If save-ref key are active and the hyperref[8] package is detected, hyperlink and hypertarget will be used, otherwise the usual "label and ref" system provided by ETEX will be used.

By design the environment cannot be nested but full supports "verbatim material" in the body and it is assumed that each numbered\item or \item* within the environment in which it is active it has a "single execution" unless \item or \item* open a nested level or use the no-store key.

The <code>anskey*</code> environment is implemented using the <code>scontents</code> package, for the correct operation <code>\begin{anskey*}</code> and <code>\end{anskey*}</code> must be in different lines, all $\langle keys \rangle$ must be passed separated by commas and "without separation" of the start of the environment. Comments "%" or "any character" after <code>\begin{anskey*}</code> or $[\langle key=val \rangle]$ on the same line are NOT supported, the package <code>scontents</code> will return an "error" message if this happens. In a similar way comments "%" or "any character" after <code>\end{anskey*}</code> on the same line the package <code>scontents</code> will return a "warning" message.

6.3.1 Keys for anskey*

The anskey* environment uses the same $\langle keys \rangle$ as the \anskey command next to the keys inherited from package scontents. The environment is available for all levels of the enumext environment and the enumext* environment, but it is disabled for the keyans, keyans* and keyanspic environments.

 $write-env = \{\langle \textit{file.ext} \rangle\}$ default: not used

Sets the name of the $\langle external\ file \rangle$ in which the $\langle contents \rangle$ of the environment will be written. The $\langle file.ext \rangle$ will be created in the working directory, relative or absolute paths are not supported. If $\langle file.ext \rangle$ does not exist, it will be created or overwritten if the overwrite key is used.

```
overwrite = \{\langle true \mid false \rangle\} default: false
```

Sets whether the $\langle \mathit{file.ext} \rangle$ generated by write-env from the anskey* environment will be rewritten.

```
force-eol = \{\langle true \mid false \rangle\} default: false
```

Sets if the *end of line* for the \(\stored \content\) is hidden or not. This key is necessary only if the last line is the closing of some environment defined by the \(\frac{fancyvrb}{fancyvrb}\) package as \\end{\verbatim}\) or another environment that does not support a comments "%" after closing \\end{\verbatim}\%.

For security reasons the keys store-env, print-env and write-out they have been left disabled. It is recommended that you review the scontents[4] documentation to understand how the keys described here work.

Example

```
\begin{enumext}[save-ans=test,show-pos=true,start=5]
  \item* Text containing our instructions or questions.
    \begin{anskey*}[item-star]
      (first answer)
    \end{anskey*}
  \item Text containing our instructions or questions.
    \begin{enumext}
      \item Question.
        \begin{anskey*}
          (second answer)
        \end{anskey*}
    \end{enumext}
  \item Text containing our instructions or questions.
    \begin{anskey*}
      (third answer)
    \end{anskey*}
  \item Text containing our instructions or questions.
    \begin{anskey*}
      (fourth answer)
    \end{anskey*}
\end{enumext}
```

- ★ 5. Text containing our instructions or questions.
- 7. Text containing our instructions or questions.
- [5] First answer with verbatim
- [7] third answer
- 6. Text containing our instructions or questions.
- 8. Text containing our instructions or questions.

(a) Question.

- [8] fourth answer
- [6] second answer

6.4 The environments keyans and keyans*

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

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

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

The $\langle keys \rangle$ set in the optional argument of the environment are the same (almost) as those of the enumext and enumext* environments and have higher precedence than those set by \setenumext[$\langle keyans \rangle$] { $\langle key = val \rangle$ } or \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*).

6.4.1 The \item* in keyans and keyans*

```
\item* \item* \item* \item* [\langle content \rangle]
```

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

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

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

Example

```
\begin{enumext}[save-ans=test,columns=2,show-ans=true]
  \item Text containing a question.
    \begin{keyans*} \[ nosep, columns=2 \]
      \item Choice
      \item* Correct choice
      \item 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

D) Choice

- C) Choice
- E) Choice

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



Some text

6.5 The environment keyanspic

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.

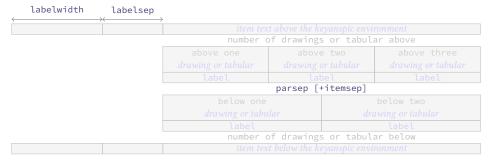


Figure 6: Representation of the keyanspic environment with optional argument [3,2] in enumext.

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.

The command \anspic

```
\anspic \anspic{\langle drawing \ or \ tabular \rangle}
                     \verb|\anspic*| | \langle \mathit{content} \rangle | \{ \langle \mathit{drawing} \ \mathit{or} \ \mathit{tabular} \rangle \}
```

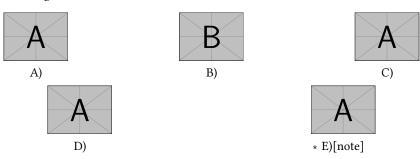
The \anspic command take three arguments, the starred argument '*' store the current \langle label \rangle next to the $\langle content \rangle$ (if it is present) in sequence and prop list $\{\langle store\ name \rangle\}$ set by save-ans key.

The starred argument '*' 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 argument '*' 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 "stored content" in prop list {\store name}} defined by save-ans key in the $\langle position \rangle$ returned by the show-pos key. The "stored content" can only be accessed after it is stored, if $\{\langle store\ name \rangle\}$ does not exist the command will return an error.

The form taken by the argument $\{\langle store\ name: position \rangle\}$ 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.

6.6.2 The command \printkeyans

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

The command \printkeyans prints "all stored content" in sequence {\store name\} defined by save-ans key placing this inside the enumext environment or the enumext* environment if the starred argument '*' is used. The "stored content" can only be accessed after it is stored in the sequence, if {\store name\}} does not exist the command will return an error.

The optional argument allows managing the $\langle keys \rangle$ in the "first level" of the environment in which the "stored content" of the sequence {\langle store name \rangle} will be printed, if the starred argument "*" is used it will be enumext* otherwise enumext.

The default values for the "first level" are the same as the default values for the enumext and enumext* environments along with the keys nosep, first=\small, font=\small and columns=2. For the inner levels of the environment enumext saved in the sequence {\store name\}} the default values are the same as those established for the second, third and fourth levels plus the keys nosep, first=\small, font=\small. If the environment enumext* is saved within the sequence $\{\langle store\ name \rangle\}$ it will have the same default values plus the keys nosep, first=\small, font=\small.

Since the command encapsulates by default the enumext environment or the enumext* environment, we must take some considerations:

- If we execute \printkeyans*{\langle store name \rangle} and the sequence {\langle store name \rangle} already contains any enumext* environment an error will be returned as we cannot nest.
- If we execute $\printkeyans*{\langle store\ name \rangle}$ and the sequence $\{\langle store\ name \rangle\}$ contains any enumext environments, they will start with the $\langle keys \rangle$ set for the first level unless they are set in the optional argument or save-key is used to modify it.
- If we execute \printkeyans{\langle store name \rangle} and the sequence {\langle store name \rangle} contains any environment enumext*, they will start with the \langle keys \rangle set by default unless they are set in the optional argument or save-key is used to modify it.

The default values for the "first level" of \printkeyans commands and \printkeyans* are established using \setenumext[\langle print, 1\rangle] \{\langle keys\rangle\} and \setenumext[\langle print*\rangle] \{\langle keys\rangle\}. If we need to set the \langle keys\rangle for the environment enumext "saved" in the sequence \{\langle store name\rangle\} we will use \setenumext[\langle print, \line level\rangle] \{\langle keys\rangle\} and if we need to set the \langle keys\rangle for the environment enumext* "saved" in the sequence \{\langle store name\rangle\} we will use \setenumext[\langle print, *\rangle] \{\langle keys\rangle\}.

Example

```
\begin{enumext} [save-ans=sample,columns=2,show-pos=true,nosep,save-ref=true]
   \item Factor 3x+3y+3z. \anskey\{3(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
                                                 (b) Rate the following package and class
[1] |3(x+y+z)|
                                                         xsim
2. True False
                                                      [4] very good
  (a) LaTeX2e is cool?
```

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
4. (b) i. very good
5. ii. obsolete
4. **
```

exsheets

[5] obsolete

ii.

7 Full examples

[2] Very True!

3. Related to Linux
(a) You use linux?

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 .

```
¹The cool TeX automation tool: https://www.ctan.org/pkg/arara
©2024 by Pablo González L
```

ı. La velocità di $1,00 \times 10$	² m/s espressa in km	/h è: 3. La velocità di 1.00×10^{-3}	² m/s espressa in km	/h è
------------------------------------	---------------------------------	---	---------------------------------	------

- 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 Å = 4. In fisica nucleare si usa l'angstrom (simbolo: 1 Å = 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?
 - A $1 \text{ Å} = 1 \times 10^5 \text{ fm}.$
 - B $1 \text{ Å} = 1 \times 10^{-5} \text{ fm}.$
 - \overline{C} 1 Å = 1 × 10⁻¹⁵ fm.
 - D $1 \text{ Å} = 1 \times 10^3 \text{ fm}.$

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

unità di misura?

A 36 km/h.

B 360 km/h.

C 27,8 km/h.

D $3.60 \times 10^8 \,\text{km/h}$.

 1×10^{-10} m) e il fermi o femtometro (1 fm =

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

- B $1 \text{ Å} = 1 \times 10^{-5} \text{ fm}.$
- \overline{C} 1 Å = 1 × 10⁻¹⁵ fm.
- D $1 \text{ Å} = 1 \times 10^3 \text{ fm}.$

3. B 1. B 2. A 4. A

Example 2

Adapted from the response given by Florent Rougon in Multiple choice questions with proposed answers in random order — addition of automatic correction (cross mark) **□**.

- 1. 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 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 ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il fermi o femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times 10^{-10} \, \text{m}$) e il femtometro ($1 \, \text{fm} = 1 \times$ 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 $1{,}00 \times 10^2 \,\mathrm{m/s}$ espressa in km/h è:
 - A 36 km/h.
- ✓ B 360 km/h.
 - C 27,8 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
- 4. A

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
 - I. $2\alpha + 2\delta = 90^{\circ}$
 - II. $\alpha = \delta$
 - III. $\angle EDF = 45^{\circ}$

- (A) I only
- (B) II only
- (C) 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:



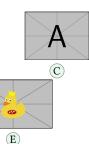
(D) I and III only

E I, II, and III

(D) value

(E) value







5. Question with image on left side:

(D)

- (A) value
- (B) value
- C value
- (D) correct
- (E) value
- Test keys



- 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 **.**



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.

Which choice best describes what happens in the passage?

A) One character argues with another charac-

- ter 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.

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.
- One character criticizes another character for pursuing an unexpected course of action.

1. A)

2. C

3. B)

4. D)

8 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= $\{\langle some \rangle\}$ 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, \textseriskcentered 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.

©2024 by Pablo González L

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 \labelbx { 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*={\labelbx{#1}}.

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[5] 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[11]</code> and <code>l3seq[11]</code> modules together with the <code>hyperref[8]</code> and <code>enumitem[6]</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 Lage X, 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.

9 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] González, Pablo. "scontents Stores LTEX contents in memory or files". Available from CTAN, https://www.ctan.org/pkg/scontents, 2022.
- [5] The LaTeX Project. "enumerate Enumerate with redefinable labels". Available from CTAN, https://www.ctan.org/pkg/enumerate, 2024.
- [6] Bezos, Javier. "Customizing lists with the enumitem package". Available from CTAN, https://www.ctan.org/pkg/enumitem, 2019
- [7] Berry, Karl. "MTX 2_{ε} : An Unofficial Reference Manual". Available from ctan, https://ctan.org/pkg/latex2e-help-texinfo, 2024.
- [8] The LTEX Project. "Extensive support for hypertext in LTEX". Available from CTAN, https://www.ctan.org/pkg/hyperref, 2024.
- [9] Burnol, Jean-François. "The footnotehyper package". Available from CTAN, https://www.ctan.org/pkg/footnotehyper, 2021.
- [10] The LTEX Project. "The expl3 package". Available from ctan, https://www.ctan.org/pkg/l3kernel, 2024.
- [11] The LTEX Project. "The LTEX3 Interfaces". Available from CTAN, https://www.ctan.org/pkg/l3kernel, 2024.
- [12] The FTeX Project. "The FTeX 2_{ε} sources". Available from CTAN, https://ctan.org/tex-archive/macros/latex/base, 2024.
- [13] The LTEX Project. "LTEX for authors current version". Available from CTAN, https://ctan.org/pkg/latex-base, 2024.
- [14] GUNDLACH, PATRICK. "The lua-visual-debug package". Available from CTAN, https://www.ctan.org/pkg/lua-visual-debug, 2023.
- [15] Lemvig, Mogens. "The shortlst package". Available from ctan, https://www.ctan.org/pkg/shortlst, 1998.
- [16] NIEDERBERGER, CLEMENS. "tasks Horizontally columned lists". Available from CTAN, https://www.ctan.org/pkg/tasks, 2022.

10 Change history

v1.0 2024-06-24 - First public release.

Index of Documentation 11

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

C	force-eol
Document class:	item-join
article	item-pos*
book	item-star
exam 2	item-sym* 13
letter 2	overwrite 14
report 2	write-env 14
\columnbreak 4, 13	Keys for environments provide by enumext:
\columnsep 10	above* 8
Commands provide by enumext:	above 8
\anskey 11-14	after 9−11
\anspic 11, 12, 15, 16	align 7, 21
\getkeyans	base-fix 8
\item* 5-7, 11, 12, 14, 15	before*9
\item 5-10, 12, 14	before 9
\miniright 10, 11	below* 9
\printkeyans 6, 12, 16	below 9
\setenumextmeta 6	check-ans 12
\setenumext 5-7, 11, 12, 14, 15, 17	columns-sep
Counters defined by enumext: enumXiii4	columns
enumXii 4	first
enumXiv 4	item-pos* 5, 6
enumXi 4	item-sym* 5,6
enumXviii	itemindent 9
enumXvii4	itemsep 8, 15
enumXvi4	labelsep 3-7, 9, 10, 12, 20, 21
enumXv 4	labelwidth 3, 4, 6, 7, 9, 10, 12, 20, 21
r.	labelwith 5
E Environments provide by enumext:	label
anskey*	list-indent 3, 9
enumext* 4-17	list-offset 3, 9, 21
enumext	listparindent 9 mark-ans 12
keyans* 4-9, 11-15	mark-pos
keyanspic 4, 7, 8, 11-15, 20	mark-ref
keyans 4-9, 11-15, 20	mini-env 4, 5, 8, 10
Environments:	mini-right* 7, 11
Verbatim14	mini-right
enumerate	mini-sep 4, 10
figure 5	no-store
list	noitemsep 8
minipage 3-5, 10, 11, 21	nosep 8, 20
multicols	overwrite
task 5	parsep 8, 15
	partopsep
F	resume* 7, 10, 11
\footnote 5	resume
I	rightmargin9
\itemsep 8	save-ans 4, 6, 10-16
(save-key 10, 11, 17
K	save-ref
Keys for \anskey provide by enumext:	save-sep11
break-col	series 7, 10, 11
item-join	show length
item-pos*	show-length
item-star	start*
Keys for anskey* provide by enumext:	start
break-col	topsep
©2024 by Pablo González L	• • • • • • • • • • • • • • • • • • • •
- 1 - 7	

widest 7	enumext 1-5, 7, 15, 22
wrap-ans12	enumitem 3-5, 9, 22
wrap-label* 8, 21	fancyvrb
wrap-label 7,8	footnotehyper
wrap-opt	hyperref 4, 5, 12, 13, 22
write-env 14	l3keys;
L	l3prop
\label4	l3seq
Labels provide by enumext:	multicol
\Alph* 7, 15	scontents 1, 2, 13, 14
\Roman* 7	task 5,0
\alph* 7	xsim
\arabic* 7	\parsep 8
\roman* 7	\partopsep
\labelsep 3, 7	
\labelwidth	R
\linewidth 10	\raggedcolumns
\listparindent 9	\ref
	\rightmargin
P	
Packages:	T
enumerate 21	\topsep

12 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: @ https://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.

12.1 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.

12.2 Initial set up

Start the DocStrip guards.

```
*package
```

Identify the internal prefix (FTEX3 DocStrip convention) for l3doc class.

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

12.3 Declaration of the package

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

```
_{\scriptscriptstyle 3} \NeedsTeXFormat{LaTeX2e}[2024-06-01]
```

Now declare the enumext package.

```
4 \ProvidesExplPackage
5 {enumext}
6 {2024-06-24}
7 {1.0}
8 {Enumerate exercise sheets}
```

Finally check if the multicol and scontents packages are loaded, if not we load it.

```
, \hook_gput_code:nnn {begindocument} {enumext}
      \IfPackageLoadedTF { multicol }
          \msg_info:nnn { enumext } { package-load } { multicol }
        }
14
15
        {
          \msg_info:nnn { enumext } { package-not-load } { multicol }
16
          \RequirePackage{multicol}[2024-05-23]
      \IfPackageLoadedTF { scontents }
        {
          \msg_info:nnn { enumext } { package-load } { scontents }
        }
23
        {
          \msg_info:nnn { enumext } { package-not-load } { scontents }
24
          \RequirePackage{scontents}
25
26
    }
27
```

12.4 Definition of variables

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

```
Integer variables will control the nesting levels of the environments and \anskey command.
       \l__enumext_level_int
     \l__enumext_level_h_int
                                 28 \int_new:N \l__enumext_level_int
\l__enumext_anskey_level_int
                                 29 \int_new:N \l__enumext_level_h_int
\l__enumext_keyans_level_int
                                 30 \int_new:N \l__enumext_anskey_level_int
                                 int_new:N \l__enumext_keyans_level_int
       \l__enumext_keyans_level_h_int
                                 32 \int_new:N \l__enumext_keyans_level_h_int
     \l__enumext_keyans_pic_level_int
                                 33 \int_new:N \l__enumext_keyans_pic_level_int
                                (End of definition for \l__enumext_level_int and others.)
                                Internal variables used by functions \__enumext_is_not_nested:, \__enumext_is_on_first_-
    \l__enumext_starred_bool
    \g__enumext_starred_bool
                                level: and \__enumext_keyans_name_and_start: (§12.5.1).
       \l__enumext_starred_first_bool
                                 34 \bool_new:N \l__enumext_starred_bool
    \l__enumext_standar_bool
                                 _{35} \bool_new:N \g__enumext_starred_bool
    \g__enumext_standar_bool
                                 _{\rm 36} \bool_new:N \l__enumext_starred_first_bool
       \l__enumext_standar_first_bool
                                 _{
m 37} \bool_new:N \l__enumext_standar_bool
                                 _{\mbox{\scriptsize $38$}} \bool_new:N \g__enumext_standar_bool
 \l__enumext_anskey_env_bool
                                 39 \bool_new:N \l__enumext_standar_first_bool
 \l__enumext_keyans_env_bool
                                 40 \bool_new:N \l__enumext_anskey_env_bool
   \g__enumext_start_line_tl
                                 \bool_new:N \l__enumext_keyans_env_bool
   \g__enumext_envir_name_tl
                                42 \tl_new:N \g__enumext_start_line_tl
   \l__enumext_envir_name_tl
                                 43 \tl_new:N
                                                \g__enumext_envir_name_tl
                                               \l__enumext_envir_name_tl
                                 44 \tl_new:N
                                (End of definition for \l_enumert_starred_bool 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. The
   \l__enumext_counter_ii_tl
                                counters enumXviii and enumXviii are used by enumext* and keyans* environments.
  \l__enumext_counter_iii_tl
   \l__enumext_counter_iv_tl
                                The initial values of these variables are set by the function \__enumext_define_counters: Nn (§12.10)
    \l__enumext_counter_v_tl
                                and then modified by the function \label{lem:node} \ and then modified by the function \ enumert_label_style: Nnn used by label key (§12.13).
   \l__enumext_counter_vi_tl
                                 45 \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 }
                                 47
                                 49 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { \__enumext_tmp:n {#1} }
                                (End of definition for \l_enumert_counter_i_tl and others.)
                                Internal variables used by ref key (§12.13).
\c__enumext_counter_style_tl
 \l__enumext_ref_key_arg_tl
                                 50 \tl_const:Nn \c__enumext_counter_style_tl
\l__enumext_ref_the_count_tl
                                    { { arabic } { roman } { Roman } { alph } { Alph } }
\l__enumext_the_counter_X_tl
                                 52 \tl_new:N \l__enumext_ref_key_arg_tl
                                 53 \tl_new:N \l__enumext_ref_the_count_tl
     \l__enumext_renew_the_count_X_tl
                                 54 \cs_set_protected:Npn \__enumext_tmp:n #1
                                 55
                                        \tl_new:c { l__enumext_renew_the_count_#1_tl }
                                        \tl_new:c { l__enumext_the_counter_#1_tl }
                                        \tl_set:ce { l__enumext_the_counter_#1_tl } { \exp_not:c { theenumX#1 } }
                                     }
                                 60 \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 (§12.24).
  \g__enumext_resume_vii_int
                                 61 \int new:N \g enumext resume int
  \l__enumext_resume_name_tl
                                 62 \int_new:N \g__enumext_resume_vii_int
       \l__enumext_resume_active_bool
                                 63 \tl_new:N
                                               \l__enumext_resume_name_tl
                                 64 \bool_new:N \l__enumext_resume_active_bool
       \g enumext starred series tl
                                 65 \tl_new:N \g__enumext_standar_series_tl
       \g__enumext_standar_series_tl
                                 66 \tl_new:N
                                               \g__enumext_starred_series_tl
```

(End of definition for \g_- enumext_resume_int and others.)

©2024 by Pablo González L 26/145

```
\l__enumext_current_widest_dim
\g__enumext_counter_styles_tl
\g__enumext_widest_label_tl
\l__enumext_label_width_by_box
```

The variable \l__enumext_current_widest_dim stores the current label width, the variable \g__enumext_counter_styles_tl stores the default $\langle label\,style \rangle$ and the variable \g__enumext_widest_label_tl the label width. These variables are used by widest (§12.14) and label (§12.12) keys.

```
67 \dim_new:N \l__enumext_current_widest_dim
68 \tl_new:N \g__enumext_counter_styles_tl
69 \tl_new:N \g__enumext_widest_label_tl
70 \box_new:N \l__enumext_label_width_by_box
```

(End of definition for \l_- enumext_current_widest_dim and others.)

\l_enumext_leftmargin_tmp_X_bool
 \l_enumext_leftmargin_tmp_X_dim
\l_enumext_leftmargin_X_dim
\l_enumext_itemindent_X_dim

```
71 \cs_set_protected:Npn \__enumext_tmp:n #1
72  {
73     \bool_new:c { l__enumext_leftmargin_tmp_#1_bool }
74     \dim_new:c { l__enumext_leftmargin_tmp_#1_dim }
75     \dim_new:c { l__enumext_leftmargin_#1_dim }
76     \dim_new:c { l__enumext_itemindent_#1_dim }
77   }
78 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { \__enumext_tmp:n {#1} }
```

(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 §12.21).

```
79 \cs_set_protected:Npn \__enumext_tmp:n #1
80  {
81     \skip_new:c { l__enumext_multicols_above_#1_skip }
82     \skip_new:c { l__enumext_multicols_below_#1_skip }
83     }
84 \clist_map_inline:nn { i, ii, iii, iv, v } { \__enumext_tmp:n {#1} }
```

 $(\textit{End of definition for } \verb|\| \verb|\| e numext_multicols_above_X_skip | and \verb|\| \verb|\| e numext_multicols_below_X_skip.)$

\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
\l__enumext_minipage_after_skip
\l__enumext_minipage_left_X_dim
\l__enumext_minipage_active_X_bool

```
%5 \int_new:N \g__enumext_minipage_stat_int
%6 \skip_new:N \l__enumext_minipage_left_skip
%7 \skip_new:N \l__enumext_minipage_right_skip
%8 \skip_new:N \g__enumext_minipage_after_skip
%9 \skip_new:N \g__enumext_minipage_right_skip
90 \skip_new:N \g__enumext_minipage_after_skip
91 \cs_set_protected:Npn \__enumext_tmp:n #1
92 {
93    \dim_new:c { l__enumext_minipage_left_#1_dim }
94    \bool_new:c { l__enumext_minipage_active_#1_bool }
95  }
96 \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 bool vars \l__enumext_wrap_label_X_bool and \l__enumext_wrap_label_opt_X_bool are used by wrap-label and wrap-label* keys ($\S12.12$), the integer \l__enumext_start_X_int are used by the start and start* keys ($\S12.14$), the token list \l__enumext_fake_item_indent_-X_tl is used by itemindent key ($\S12.17.1$), the variables \l_enumext_label_fill_left_X_tl and \l_enumext_label_fill_left_X_tl are used by the align key ($\S12.12$). 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 ($\S12.19$).

```
97 \cs_set_protected:Npn \__enumext_tmp:n #1
98
       \bool_new:c { l__enumext_wrap_label_#1_bool
       \bool_new:c { l__enumext_wrap_label_opt_#1_bool }
100
       \int_new:c { l__enumext_start_#1_int
101
       \tl_new:c
                  { l__enumext_fake_item_indent_#1_tl }
102
                   { l__enumext_label_fill_left_#1_tl
       \tl new:c
103
                   { l__enumext_label_fill_right_#1_tl }
       \bool_new:c { l__enumext_vspace_a_star_#1_bool
105
       \bool_new:c { l__enumext_vspace_b_star_#1_bool }
108 \clist_map_inline:nn { i, ii, iii, iv, v, vii, viii } { \__enumext_tmp:n {#1} }
©2024 by Pablo González L
```

27 / 145

```
(End of definition for \l_enumext_wrap_label_X_bool and others.)
       \l__enumext_store_active_bool
                                 The variable \l_enumext_store_active_bool setting by save-ans key (§12.25.1) activates all the
                                  mechanism related to \anskey, anskey*, keyans, keyans* and keyanspic environments.
  \l__enumext_store_name_tl
  \g__enumext_store_name_tl
                                  The variable \l_enumext_store_name_tl saves the \{\langle store\ name \rangle\} set by the save-ans key of the
     \l__enumext_store_anskey_arg_tl
                                  \textit{sequence} \ \text{and} \ \textit{prop list} \ \text{in which we will store, the variable} \ \\ \setminus \texttt{g\_enumext\_store\_name\_tl} \ \text{it's just a global}
     \l__enumext_store_anskey_env_tl
                                  copy of \{\langle store\ name \rangle\} used by different functions.
     \l__enumext_store_anskey_opt_tl
                                  The variable \l__enumext_store_anskey_arg_tl save the argument of \anskey (§12.29) and the
   \l__enumext_store_current_label_tl
                                  variables \l__enumext_store_anskey_env_tl and \l__enumext_store_anskey_opt_tl save the
 \l__enumext_store_current_opt_arg_tl
                                  \langle body \rangle and the \langle keys \rangle of the environment anskey* (§12.30).
\l__enumext_store_current_label_tmp_tl
                                  The variables \l__enumext_store_current_label_tl and \l__enumext_store_current_opt_-
                                  arg_tl save the current label and optional argument of \item* (§12.36) and \anspic* (§12.40.1) for the
                                  keyans, keyans* and keyanspic environments.
                                  The variable \l__enumext_store_current_label_tmp_tl is a temporary variable used by keyans,
                                  keyans* and keyanspic at various points.
                                  \bool_new:N \l__enumext_store_active_bool
                                  110 \tl new:N
                                                  \l__enumext_store_name_tl
                                  111 \tl_new:N
                                                  \g__enumext_store_name_tl
                                  112 \tl_new:N
                                                  \l__enumext_store_anskey_arg_tl
                                  113 \tl_new:N
                                                  \l__enumext_store_anskey_env_tl
                                                  \l__enumext_store_anskey_opt_tl
                                  114 \tl_new:N
                                  115 \tl_new:N
                                                   \l__enumext_store_current_label_tl
                                  116 \tl_new:N
                                                   \l__enumext_store_current_opt_arg_tl
                                  117 \tl_new:N
                                                   \l__enumext_store_current_label_tmp_tl
                                  (\textit{End of definition for} \ \backslash \ l\_\texttt{enumext\_store\_active\_bool} \ \textit{ and others.})
                                 Internal variables used by the command \setenumext (§12.47).
 \l__enumext_setkey_tmpa_tl
 \l__enumext_setkey_tmpb_tl
                                  118 \tl_new:N \l__enumext_setkey_tmpa_tl
\l enumext setkey tmpa int
                                  \tl_new:N \l__enumext_setkey_tmpb_tl
\l__enumext_setkey_tmpa_seq
                                  \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 the \printkeyans command (\§12.46) and \foreachkeyans command (\§12.49).
   \l__enumext_meta_path_tl
       \l__enumext_foreach_print_seq
                                  123 \tl_new:N \l__enumext_meta_path_tl
     \l__enumext_foreach_name_prop_tl
                                  \seq_new:N \l__enumext_foreach_print_seq
  \g__enumext_foreach_default_keys_tl
                                  125 \tl_new:N \l__enumext_foreach_name_prop_tl
                                  126 \tl_new:N \g__enumext_foreach_default_keys_tl
                                 (End of definition for \l_enumert_meta_path_tl and others.)
  \l_enumext_print_keyans_starred_tl Internal variables used by command \printkeyans (\$12.46), show-pos key (\$12.26), item-sym* key
                                 (§12.34), save-key key (§12.26.2) and "storage level system".
       \l__enumext_mark_position_str
      \g__enumext_item_symbol_aux_tl
                                  \label{eq:local_local_local_local} $$ $$ \t = N \ \l_enumext\_print_keyans\_starred\_tl $$
       \l__enumext_print_keyans_X_tl
                                  \str_new:N \l__enumext_mark_position_str
                                  129 \tl_new:N \g__enumext_item_symbol_aux_tl
     \l enumext store save kev X tl
                                  \cs_set_protected:Npn \__enumext_tmp:n #1
    \l__enumext_store_save_key_X_bool
 \l__enumext_store_upper_level_X_bool
                                  131
                                                        { l__enumext_print_keyans_#1_tl
                                  132
                                          \tl new:c
                                          \tl_new:c
                                                       { l__enumext_store_save_key_#1_tl
                                  133
                                          \bool_new:c { l__enumext_store_save_key_#1_bool
                                          \bool_new:c { l__enumext_store_upper_level_#1_bool }
                                  135
                                  136
                                  \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 used by keyanspic environment (§12.40.2).
     \l__enumext_keyans_pic_body_seq
     \l__enumext_keyans_pic_width_dim
                                  \seq_new:N \l__enumext_keyans_pic_body_seq
                                  _{\rm 139} \dim_new:N \l__enumext_keyans_pic_width_dim
     \l__enumext_keyans_pic_above_int
```

\int_new:N \l__enumext_keyans_pic_above_int

'int_new:N \l__enumext_keyans_pic_below_int

\skip_new:N \l__enumext_keyans_pic_above_skip

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

\l enumext kevans pic below int

\l__enumext_keyans_pic_above_skip

©2024 by Pablo González L 28/145

```
\l_enumext_check_answers_bool Internal variables used by "internal check answer" mechanism (§12.25.3) used by the check-ans and
                                              no-store keys and check for starred commands \item* in keyans and keyans* environments and
          \g__enumext_check_ans_key_bool
                                              \anspic* in keyanspic environment.
    \l__enumext_check_start_line_env_tl
      \g__enumext_check_starred_cmd_int
                                               \bool_new:N \l__enumext_check_answers_bool
 \g__enumext_item_anskey_int
                                               \bool_new:N \g__enumext_check_ans_key_bool
 \g__enumext_item_number_int
                                               \tl_new:N \l__enumext_check_start_line_env_tl
                                               \int_new:N \g__enumext_check_starred_cmd_int
\g__enumext_item_number_bool
                                               \int_new:N \g__enumext_item_anskey_int
        \g__enumext_item_answer_diff_int
                                               148 \int_new:N \g__enumext_item_number_int
                                               _{\mbox{\scriptsize 149}} \bool_new:N \l__enumext_item_number_bool
                                               _{150} \int_new:N \g_enumext_item_answer_diff_int
                                              (\textit{End of definition for } \verb|\l_enumext_check_answers_bool| \textit{ and others.})
                                              The boolean variable \l_enumext_hyperref_bool will determine if the hyperref package is present
    \l__enumext_hyperref_bool
                                              or load in memory (§12.8). The boolean variable \load{ll}_{-}enumext_footnotes_key_bool determine if
          \l__enumext_footnotes_key_bool
                                               hyperref is load with key hyperfootnotes=true.
                                               \text{\lool_new:N \l__enumext_hyperref_bool}
                                               152 \bool_new:N \l__enumext_footnotes_key_bool
                                              (\textit{End of definition for } \ | \ l\_enumext\_hyperref\_bool \ \ and \ | \ l\_enumext\_footnotes\_key\_bool.)
                                              Internal variables used by save-ref key (§12.26). The variables \l__enumext_label_copy_X_tl cor-
         \l__enumext_newlabel_arg_one_tl
                                              respond to temporary copies of the (labels) defined by level on which operations will be performed.
         \l__enumext_newlabel_arg_two_tl
          \l__enumext_write_aux_file_tl
                                              \l__enumext_label_copy_X_tl
                                              be used to form the arguments passed to the function \__enumext_newlabel:nn (§12.8) and the variable
                                               \l__enumext_write_aux_file_tl will be in charge of executing the writing code in the .aux file.
                                               153 \tl_new:N \l__enumext_newlabel_arg_one_tl
                                               154 \tl_new:N \l__enumext_newlabel_arg_two_tl
                                               155 \tl_new:N \l__enumext_write_aux_file_tl
                                               156 \cs_set_protected:Npn \__enumext_tmp:n #1
                                                          \tl_new:c { l__enumext_label_copy_#1_tl }
                                               158
                                               '160 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { \__enumext_tmp:n {#1} }
                                              (End of definition for \lower l=lower l=lowe
                                              Internal variables used for redefinition of \footnote (§12.42).
      \g__enumext_footnote_int
\g__enumext_footnote_arg_seq
                                               'int_new:N \g__enumext_footnote_int
\g__enumext_footnote_int_seq
                                               \seq_new:N \g__enumext_footnote_arg_seq
                                               \seq_new:N \g__enumext_footnote_int_seq
                                              seq.)
                                              Internal variables used by enumext* and keyans* environments.
        \l__enumext_item_starred_X_bool
        l__enumext_item_column_pos_X_int
                                               164 \cs_set_protected:Npn \__enumext_tmp:n #1
        \g__enumext_item_count_all_X_int
                                               165 {
           \l__enumext_joined_item_X_int
                                                          \bool_new:c { l__enumext_item_starred_#1_bool
                                                          \int_new:c { l__enumext_item_column_pos_#1_int }
       \l__enumext_joined_item_aux_X_int
                                                          \int_new:c { g__enumext_item_count_all_#1_int
         \l__enumext_tmpa_X_int
                                                          \int_new:c { l__enumext_joined_item_#1_int
         \l__enumext_tmpa_X_dim
                                                          \int_new:c { l__enumext_joined_item_aux_#1_int
 \l__enumext_item_text_X_box
                                                          \int_new:c { l__enumext_tmpa_#1_int
         \l__enumext_joined_width_X_dim
                                                          \dim_new:c { l__enumext_tmpa_#1_dim
                                               172
\l__enumext_item_width_X_dim
                                                          \box_new:c { l__enumext_item_text_#1_box
                                               173
       \g__enumext_item_symbol_aux_X_tl
                                                          \dim_new:c { l__enumext_joined_width_#1_dim
                                               174
          \l__enumext_align_label_X_str
                                                          \dim_new:c { l__enumext_item_width_#1_dim
                                               175
     \g__enumext_minipage_active_X_bool
                                                          \tl_new:c { g__enumext_item_symbol_aux_#1_tl
       \l__enumext_miniright_code_X_box
                                                          \str_new:c { l__enumext_align_label_#1_str
     \g__enumext_minipage_center_X_bool
                                                          \bool_new:c { g__enumext_minipage_active_#1_bool }
       \g__enumext_minipage_right_X_dim
                                                          \box_new:c { l__enumext_miniright_code_#1_box
                                                          \bool_new:c { g__enumext_minipage_center_#1_bool }
       \g__enumext_minipage_right_X_skip
                                               180
                                                         \dim_new:c { g__enumext_minipage_right_#1_dim
                                               181
                                                         \skip_new:c { g__enumext_minipage_right_#1_skip }
                                               182
                                               183
                                               184 \clist_map_inline:nn { vii, viii } { \__enumext_tmp:n {#1} }
                                              (End of definition for \l_enumext_item_starred_X_bool and others.)
```

©2024 by Pablo González L

29 / 145

```
\c__enumext_all_envs_clist An internal clist-var variable to run with \__enumext_tmp:n.
                              185 \clist_const:Nn \c__enumext_all_envs_clist
                              186
                                  {
                                     {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv},
                              187
                                     {keyans}{v}, {enumext*}{vii}, {keyans*}{viii}
                              188
                              189
                             (End of definition for \c_enumext_all_envs_clist.)
                              12.5 Some utility functions
      \keys_precompile:neN
                             Non-standard kernel variants used by the \printkeyans command (§12.46) and \foreachkeyans com-
               \seq_use:NV
                             mand (§12.49).
                              \cs_generate_variant:Nn \keys_precompile:nnN { neN }
                              \cs_generate_variant:Nn \seq_use:Nn { NV }
                             (End of definition for \keys_precompile:neN and \seq_use:NV.)
      _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 }
                              194
                                   }
                              195
                             (End of definition for \_=enumext_at_begin_document:n.)
                             A internal "hook" functions for execute code mini-right and mini-right* keys outside the enumext*
   \__enumext_after_env:nn
  \ enumext before env:nn
                             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}
                                  }
                              \cs_new_protected:Npn \__enumext_before_env:nn #1 #2
                                     \hook_gput_code:nnn {env/#1/before} {enumext} {#2}
                                   }
                             (\textit{End of definition for } \verb|\|\_enumext\_after\_env:nn| \  \, \textit{and } \verb|\|\_enumext\_before\_env:nn.)
         \__enumext_level: Function for check current level in enumext.
                              204 \cs_new:Nn \__enumext_level:
                                 {
                                     \int_to_roman:n { \l__enumext_level_int }
                                   }
                              207
                             (End of definition for \__enumext_level:.)
   \__enumext_if_is_int:nT 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
   \__enumext_if_is_int:nF
  \__enumext_if_is_int:nTF
                             function argument is an integer expression?.
                              208 \prg_new_protected_conditional:Npnn \__enumext_if_is_int:n #1 { T, F, TF }
                                  {
                                     \regex_match:nnTF { ^[\+\-]?[\d]+$ } {#1} % $
                                       { \prg_return_true: }
                                       { \prg_return_false: }
                                   }
                             (\textit{End of definition for } \_\_enumext\_if\_is\_int:nT, \\ \_\_enumext\_if\_is\_int:nF.)
                             \__enumext_regex_counter_style:
                             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 \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
                              216
```

\regex_replace_once:nnN { \c{##1}* }

221 }

©2024 by Pablo González L 30 / 145

{ \c{##1}\cB{\u{l_enumext_ref_the_count_tl}\cE} } \l_enumext_ref_key_arg_tl

(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.

(End of definition for $__$ enumext $_$ show $_$ length:nnn.)

12.5.1 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 and save the environment name in \l__enumext_envir_name_tl.

```
\cs_new_protected:Nn \__enumext_is_not_nested:
229
      \str_case:en { \@currenvir }
230
        {
231
          {enumext}
               \tl_set:Nn \l__enumext_envir_name_tl { enumext }
               \verb|\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
                 }
             }
           {enumext*}
               \tl_set:Nn \l__enumext_envir_name_tl { enumext* }
               \bool_lazy_and:nnT
                 { \bool_not_p:n { \g__enumext_starred_bool } }
                 { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
                 {
                   \bool_gset_true:N \g__enumext_starred_bool
                 }
251
        }
252
```

The function __enumext_is_on_first_level: will set the variables \l__enumext_standar_first_bool ($\S12.25.1$), \l__enumext_starred_first_bool ($\S12.25.1$) and \l__enumext_anskey_env_bool ($\S12.30$) 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.

```
254 \cs_new_protected:Nn \__enumext_is_on_first_level:
    {
255
       \bool_lazy_all:nT
256
         {
257
           { \bool_if_p:N \g__enumext_standar_bool }
258
           { \int_compare_p:nNn { \l__enumext_level_int } = { 1 } }
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
         }
           \bool_set_true:N \l__enumext_standar_first_bool
           \bool_set_true:N \l__enumext_anskey_env_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
           { \bool_if_p:N \g__enumext_starred_bool }
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
©2024 by Pablo González L
```

```
{ \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
         }
276
         {
            \bool_set_true:N \l__enumext_starred_first_bool
278
            \bool_set_true:N \l__enumext_anskey_env_bool
            \tl_gset:Nn \g__enumext_envir_name_tl { enumext* }
            \tl_gset:Ne \g__enumext_start_line_tl
                on ~ line ~ \exp_not:V \inputlineno
         }
     }
(End of definition for \__enumext_is_not_nested: and \__enumext_is_on_first_level:.)
```

__enumext_keyans_name_and_start:

The function __enumext_keyans_name_and_start: will save the start line number and name of the environments keyans, keyans* and keyanspic in the variables \l__enumext_check_start_line_env_tl and \l__enumext_envir_name_tl to use in the __enumext_check_starred_cmd:n function.

```
287 \cs_new_protected:Nn \__enumext_keyans_name_and_start:
    {
288
      \str_case:en { \@currenvir }
289
        {
290
           {keyans}
            {
               \tl_set:Nn \l__enumext_envir_name_tl { keyans }
               \tl_set:Ne \l__enumext_check_start_line_env_tl
                 {
                   in ~ 'keyans' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
            }
           {keyans*}
            {
               \tl_set:Nn \l__enumext_envir_name_tl { keyans* }
               \tl_set:Ne \l__enumext_check_start_line_env_tl
                {
                   in ~ 'keyans*' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
            }
           {keyanspic}
            {
               \tl_set:Nn \l__enumext_envir_name_tl { keyanspic }
               \tl_set:Ne \l__enumext_check_start_line_env_tl
                 {
                   in ~ 'keyanspic' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
                 }
            }
        }
    }
```

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

12.5.2 Utilities for log and terminal

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

```
\__enumext_reset_global_int:
       \__enumext_reset_global_bool:
\__enumext_reset_global_tl:
```

```
\__enumext_reset_global_int:
      \__enumext_reset_global_bool:
      \__enumext_reset_global_tl:
    }
322
323 \cs_new_protected:Nn \__enumext_reset_global_int:
324
      \int_gzero:N \g__enumext_item_number_int
325
      \int_gzero:N \g__enumext_item_anskey_int
326
      \int_gzero:N \g__enumext_item_answer_diff_int
327
328
329 \cs_new_protected:Nn \__enumext_reset_global_bool:
      \bool_gset_false:N \g__enumext_check_ans_key_bool
      ©2024 by Pablo González L
```

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

__enumext_log_global_vars:
__enumext_log_answer_vars:

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.

 $(\textit{End of definition for } \verb|_enumext_log_global_vars: and \verb|_enumext_log_answer_vars:|)$

12.6 Copying list and minipage environments

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

```
\label{eq:cont_arg_one} $$ \left( arg one \right) \left\{ \left( arg two \right) \right\} $$ \left( opt \right) $$ \end{supersolution} $$ \end{supersolution} $$ \left( opt \right) $$ \end{supersolution} $$ \end{supersolution} $$ \left( opt \right) $$ \end{supersolution} $$ \end{supersolution} $$ \left( opt \right) $$ \end{supersolution} $$ \end{supersolution} $$ \left( opt \right) $$ \end{supersolution} $$ \left( opt \right) $$ \end{supersolution} $$
```

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.

```
356 \__enumext_at_begin_document:n
357 {
358      \cs_new_eq:NN \__enumext_start_list:nn \list
359      \cs_new_eq:NN \_enumext_stop_list: \endlist
360      \cs_new_eq:NN \__enumext_item_std:w \item
361 }
```

(End of definition for __enumext_start_list:nn, __enumext_stop_list:, and __enumext_item_std:w.)
The minipage environment provided by LTEX has the following (simplified) plain form:

```
\label{eq:linear-pos} $$ \min_{\substack{c \in \{pos\} \ | \{\langle height \rangle \ | \{\langle inner-pos \rangle \ | \{\langle width \rangle \} \\ \langle internal \ implement \rangle \\ \\ endminipage} $$
```

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.

©2024 by Pablo González L 33 / 145

12.7 The internal minipage environment

__enumext_internal_mini_page:
 __enumext_mini_env*

The function __enumext_internal_mini_page: creates a internal __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 \skip_vertical:N \c_zero_skip to maintain alignment on "top" in the first part and \skip_vertical:N \c_zero_skip in the second part to allow spaces "below". 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. This function is passed to the function __enumext_safe_exec: in the enumext environment definition (§12.38) and __enumext_safe_exec_vii: in the enumext* environment definition (§12.43)

```
367 \cs_new_protected:Nn \__enumext_internal_mini_page:
    {
368
      \int_compare:nNnT { \l__enumext_level_int } = { 0 }
369
370
           \DeclareDocumentEnvironment{__enumext_mini_env*}{ m }
371
             {
               \__enumext_minipage:w [ t ] { ##1 }
                 \legacy_if_gset_false:n { @minipage }
                 \skip_vertical:N \c_zero_skip
             }
             {
                 \skip_vertical:N \c_zero_skip
378
               \__enumext_endminipage:
379
         }
381
```

(End of definition for __enumext_internal_mini_page: and __enumext_mini_env*.)

12.8 Compatibility with hyperref and footnotehyper

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

```
_{383} \rightarrow \{ \color{local} \co
```

__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.

```
bool_if:NT \l__enumext_footnotes_key_bool

{

vifPackageLoadedTF { footnotehyper }

{

vmsg_info:nnn { enumext } { package-load } { footnotehyper }

}

{

typeout{No ~ footnotehyper ~ load}

vtypeout{Load ~ and ~ use ~ \string\makesavenoteenv{enumext*}}

bool_set_false:N \l__enumext_footnotes_key_bool
}

}
```

©2024 by Pablo González L

34 / 145

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.

 $(\textit{End of definition for } \verb|\|_enumext_after_hyperref:|, \verb|\|_enumext_hypertarget:|n|, and \verb|\|_enumext_phantomsection:|)$

__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.

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

12.9 Definition of public dimension

The package enumext only provides a single public dimension \itemwidth and is intended for user convenience only and is not for internal use as such. This dimension is set in all environments and is only used by the wrap-ans key at its default value.

```
_{436} \dim_zero_new:N \itemwidth
```

12.10 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:

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

#2: The counter's name.

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

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
         446 \__enumext_define_counters:Nn \l__enumext_counter_i_tl { enumXi
  enumXv
         __enumext_define_counters:Nn \l__enumext_counter_ii_tl { enumXii
 enumXvi 448 \__enumext_define_counters:Nn \l__enumext_counter_iii_tl { enumXiii }
enumXvii 449 \__enumext_define_counters:Nn \l__enumext_counter_iv_tl { enumXiv }
enumXviii 450 \__enumext_define_counters:Nn \l__enumext_counter_v_tl { enumXv
         451 \__enumext_define_counters:Nn \l__enumext_counter_vi_tl { enumXvi
         453 \__enumext_define_counters:Nn \l__enumext_counter_viii_tl { enumXviii }
```

(End of definition for enumXi and others.)

12.11 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 $\langle counters \rangle$ will be used as default $\langle labels \rangle$ 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.

```
454 \cs_new_protected:Npn \__enumext_register_counter_style:Nn #1 #2
455 {
      \tl_const:cn { c__enumext_widest_ \cs_to_str:N #1 _tl } {#2}
456
      \tl_gput_right:Nn \g__enumext_counter_styles_tl {#1}
457
458
459 \__enumext_register_counter_style:Nn \arabic { 0 }
460 \__enumext_register_counter_style:Nn \Alph { M }
461 \__enumext_register_counter_style:Nn \alph { m }
462 \__enumext_register_counter_style:Nn \Roman { VIII }
463 \__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.

```
464 \cs_new_protected:Npn \__enumext_label_width_by_box:Nn #1 #2
465 {
      \hbox_set:Nn \l__enumext_label_width_by_box {#2}
      \dim_set:Nn #1 { \box_wd:N \l__enumext_label_width_by_box }
   }
469 \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.

```
470 \cs_new_protected:Npn \__enumext_label_style:Nnn #1 #2 #3
471
      \tl_clear_new:N #1
472
      \tl_put_right:Ne #1 { \tl_trim_spaces:n {#3} }
473
      \tl_gset_eq:NN \g__enumext_widest_label_tl #1
474
      \tl_map_inline:Nn \g__enumext_counter_styles_tl
475
        {
476
          \tl_replace_all:Nne #1 { ##1* } { \exp_not:N ##1 {#2} }
477
          \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
481
        { \tl_use:N \g__enumext_widest_label_tl }
      \tl_set_eq:cN { the #2 } #1
483
485 \cs_generate_variant:Nn \__enumext_label_style:Nnn { cvn }
```

©2024 by Pablo González L

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

12.12 Setting keys associated with label

Definition of keys font, labelsep, labelwidth, wrap-label and wrap-label* keys for enumext and keyans environments. labelsep labelwidth 486 \cs_set_protected:Npn __enumext_tmp:nn #1 #2 wrap-label 487 wrap-label* \keys_define:nn { enumext / #1 } 488 { font .tl_set:c = { l__enumext_label_font_style_#2_tl }, .value_required:n = true, font labelsep .dim_set:c = { l__enumext_labelsep_#2_dim }, labelsep .initial:n = {0.3333em}, labelsep .value_required:n = true, labelwidth .dim_set:c = { l__enumext_labelwidth_#2_dim }, labelwidth .value_required:n = true, wrap-label .cs_set_protected:cp = { __enumext_wrapper_label_#2:n } ##1, 497 wrap-label .initial:n = {##1}, .value_required:n = true, wrap-label wrap-label* .code:n = { \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, } 507 \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. The align key is implemented differently for "starred" and "non starred" environments. 508 \cs_set_protected:Npn __enumext_tmp:nn #1 #2 { \keys_define:nn { enumext / #1 } 511 { align .choice:, align / left .code:n = { \tl_clear:c { l__enumext_label_fill_left_#2_tl } \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill } 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 = 523 { \tl_set:cn { l__enumext_label_fill_left_#2_tl } { \hfill } \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill } }, align / unknown .code:n = \msg_error:nneee { enumext } { unknown-choice } { align } { left, ~ right, ~ center } { \exp_not:n {##1} }, align .initial:n = left, 531 align .value_required:n = true, 532 } 534 535 \clist_map_inline:nn 536 {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {keyans}{v} 537 538 { __enumext_tmp:nn #1 }

540 \cs_set_protected:Npn __enumext_tmp:nn #1 #2

\keys_define:nn { enumext / #1 }

align .choice:,

©2024 by Pablo González L

541

543

(End of definition for align.)

12.13 Setting label and ref keys

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 $\l_enumext_label_X$

12.13.1 Define and set label and ref keys for enumext environment

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

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

label

```
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
                            1.
          label .initial:n = #3,
566
          label .value_required:n = true,
567
          ref
                .code:n
                         = \__enumext_standar_ref:n {##1},
          ref
                .value_required:n = true,
571
573 \__enumext_tmp:nnn { level-2 } { ii } { (\alph*) }
574 \__enumext_tmp:nnn { level-3 } { iii } { \roman*. }
575 \__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 *empty* 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.

```
576 \cs_new_protected:Npn \__enumext_standar_ref:n #1
577
      \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
578
      \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
579
        {
          \msg_error:nnn { enumext } { key-ref-empty } { enumext }
        }
          \tl set eq:Nc
            \l__enumext_ref_the_count_tl { l__enumext_counter_ \__enumext_level: _tl }
          \__enumext_regex_counter_style:
          \tl set ea:Nc
587
            \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 }
        }
    7
```

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

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

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

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

\l__enumext_label_viii_tl

```
603 \cs_set_protected:Npn \__enumext_tmp:nnn #1 #2 #3
604
      \keys_define:nn { enumext / #1 }
605
        {
          label .code:n
                              \__enumext_label_style:cvn { l__enumext_label_#2_tl }
                                 \dim_set_eq:cN { l__enumext_labelwidth_#2_dim }
                                \l__enumext_current_widest_dim
          label .initial:n = #3,
          label .value_required:n = true,
614
               .code:n = \__enumext_starred_ref:n {##1},
                .value_required:n = true,
          ref
616
        }
617
619 \__enumext_tmp:nnn { enumext* } { vii } { \arabic*.}
620 \__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.

```
621 \cs_new_protected:Npn \__enumext_starred_ref:n #1
622
       \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
623
       \int_compare:nNnT { \l__enumext_level_h_int } = { 1 }
624
625
           \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
626
             {
               \msg_error:nnn { enumext } { key-ref-empty } { enumext* }
             }
               \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_vii_tl
               \__enumext_regex_counter_style:
               \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 }
                 }
638
             }
       \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
         {
           \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
             {
               \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
               \tl_put_right:Ne \l__enumext_renew_the_count_viii_tl
                   \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
©2024 by Pablo González L
```

39 / 145

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.

(End of definition for __enumext_starred_ref:n and __enumext_starred_ref:.)

12.13.3 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
                          676 \keys_define:nn { enumext / keyans }
\l__enumext_label_vi_tl
                              {
                          677
                                                  = {
                                label .code:n
                          678
                                                      \__enumext_label_style:cvn { l__enumext_label_v_tl }
                          679
                                                        { l__enumext_counter_v_tl } {#1}
                                                      \dim_set_eq:cN { l__enumext_labelwidth_v_dim }
                                                        \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*),
                                label .value_required:n = true,
                                ref
                                       .code:n
                                                = \__enumext_keyans_ref:n {#1},
```

.value_required:n = true,

(End of definition for label and others.)

ref

692 }

__enumext_keyans_ref:n
__enumext_keyans_ref:

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

```
693 \cs_new_protected:Npn \__enumext_keyans_ref:n #1
694
      \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
695
      \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
696
        {
          \msg_error:nnn { enumext } { key-ref-empty } { keyans }
        }
        {
          \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
            {
               \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl }
                 { \exp_not:V \l__enumext_ref_key_arg_tl }
        }
    }
```

©2024 by Pablo González L

40 / 145

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

(End of definition for __enumext_keyans_ref:n and __enumext_keyans_ref:.)

12.14 Setting start, start* and widest keys

__enumext_start_from:NNn
__enumext_start_from:ccn
__enumext_start_from:cce

The function $_$ enumext_start_from: NNn used by start and start* keys 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 \Alph, \alph, \Roman or \roman. This effectively allows start=A or start=1 to be used.

```
718 \cs_new_protected:Npn \__enumext_start_from:NNn #1 #2 #3
719
         _enumext_if_is_int:nTF { #3 }
720
          {
            \int_set:Nn #2 {#3}
         }
          {
            \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} } }
728
729
730
731 \cs_generate_variant:Nn \__enumext_start_from:NNn { ccn, cce }
```

(End of definition for $_$ enumext_start_from:NNn.)

__enumext_widest_from:nNNn
__enumext_widest_from:nccn

widest

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.

```
732 \cs_new_protected:Npn \__enumext_widest_from:nNNn #1 #2 #3 #4
733
       \__enumext_if_is_int:nTF {#4}
734
735
           \setcounter{enumX#1} { #4 }
        }
        {
           \regex_match:nVT { \c{Alph} | \c{alph} } {#2}
             { \setcounter{enumX#1} { \int_from_alph:n {#4} } }
           \regex_match:nVT { \c{Roman} | \c{roman} } {#2}
             { \setcounter{enumX#1} { \int_from_roman:n {#4} } }
743
        \__enumext_label_width_by_box:cv
744
          { l__enumext_labelwidth_#1_dim } { l__enumext_label_#1_tl }
745
746
    }
747 \cs_generate_variant:Nn \__enumext_widest_from:nNNn { nccn }
```

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

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

```
748 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
749 {
©2024 by Pablo González L
```

```
\keys_define:nn { enumext / #1 }
750
        {
751
          start .code:n
                                 \ enumext start from:ccn
                                   { l__enumext_label_#2_tl }
                                   { l__enumext_start_#2_int } {##1}
                               1.
          start .initial:n = 1,
          start .value_required:n = true,
          start* .code:n
                            = {
                                 \__enumext_start_from:cce
                                   { l__enumext_label_#2_tl }
                                   { l__enumext_start_#2_int } { \int_eval:n {##1} }
                               1.
          start* .value_required:n = true,
                             = {
          widest .code:n
765
                                 \__enumext_widest_from:nccn {#2}
                                   { l__enumext_label_#2_tl }
                                   { l__enumext_labelwidth_#2_dim } {##1}
                               },
          widest .value_required:n = true,
773 \clist_map_inline:Nn \c_enumext_all_envs_clist { \_enumext_tmp:nn #1 }
```

(End of definition for start, start*, and widest.)

12.15 Setting keys for vertical spaces

```
Define and set topsep, partopsep, parsep, itemsep, noitemsep and nosep keys for enumext,
   topsep
           enumext*, keyans and keyans* environments.
partopsep
   parsep
           774 \cs_set_protected:Npn \__enumext_tmp:nnnnnn #1 #2 #3 #4 #5 #6
noitemsep
           775
    nosep
                  \keys_define:nn { enumext / #1 }
           776
                      topsep
                                .skip_set:c = { l__enumext_topsep_#2_skip },
           778
                                .initial:n = {#3},
                      topsep
           779
                      topsep
                                .value_required:n = true,
                      partopsep .skip_set:c = { l__enumext_partopsep_#2_skip },
                      partopsep .initial:n = {#4},
                      partopsep .value_required:n = true,
           783
                              .skip_set:c = { l__enumext_parsep_#2_skip },
                      parsep
           784
                                .initial:n = \{\#5\},
                      parsep
                                .value_required:n = true,
                      parsep
                      itemsep .skip_set:c = { l__enumext_itemsep_#2_skip },
                      itemsep .initial:n = {#6},
                      itemsep
                               .value_required:n = true,
                      noitemsep .meta:n = { itemsep = Opt, parsep = Opt },
                      noitemsep .value_forbidden:n = true,
                      nosep
                                .meta:n
                                                 itemsep = 0pt, parsep= 0pt,
                                                 topsep = 0pt, partopsep = 0pt,
           794
                                               1.
           795
                                 .value_forbidden:n = true,
                      nosep
           796
           797
```

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

(End of definition for topsep and others.)

12.16 Setting base-fix key

When nesting starting right after \item (without material between them) there is a problem with the alignment of the baseline between the two environments. One way to get around this problem is to place \mode_leave_vertical: and then apply \vspace{-\baselineskip} and set topsep=0pt for the "first level" of the nested enumext or enumext* environments.

__enumext_nested_base_line_fix:

We define the key base-fix only for the "first level" of enumext and enumext*.

The function __enumext_nested_base_line_fix: will be in charge of applying the baseline correction and adjusting the $\langle keys \rangle$. This function is passed to the function __enumext_parse_keys:n in the enumext environment definition ($\S12.38$) and to the function __enumext_parse_keys_vii:n in the enumext* environment definition ($\S12.43$)

```
828 \cs_new_protected:Nn \__enumext_nested_base_line_fix:
   {
829
      \bool_lazy_and:nnT
830
        { \bool_if_p:N \l__enumext_standar_first_bool }
831
        { \bool_if_p:N \l__enumext_base_line_fix_bool }
832
        {
833
           \mode_leave_vertical:
834
           \vspace { -\baselineskip }
835
           \keys_set:nn { enumext / level-1 }
             {
               topsep = Opt, above = Opt, above* = Opt,
             }
        }
      \bool_lazy_and:nnT
841
        { \bool_if_p:N \l__enumext_starred_first_bool }
842
        { \bool_if_p:N \l__enumext_base_line_fix_bool }
843
           \mode_leave_vertical:
           \vspace { -\baselineskip }
           \keys_set:nn { enumext / enumext* }
               topsep = Opt, above = Opt, above* = Opt,
850
851
       \bool_set_false:N \l__enumext_base_line_fix_bool
852
853
```

This key is enabled by default in the command \printkeyans (§12.46).

(End of definition for base-fix and __enumext_nested_base_line_fix:.)

12.17 Setting keys for horizontal spaces

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

.value_required:n = true,

.dim_set:c = { l__enumext_rightmargin_#2_dim },

©2024 by Pablo González L

itemindent rightmargin

```
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 }
                          \dim_set:cn { l__enumext_leftmargin_tmp_#2_dim } {##1},
          list-indent
                        .value_required:n = true,
        }
871
872 \clist_map_inline:nn
873
   {
      {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {keyans}{v}
874
875
    { \__enumext_tmp:nn #1 }
```

(End of definition for itemindent and others.)

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

```
877 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
878
      \keys_define:nn { enumext / #1 }
879
        {
          itemindent
                      .dim_set:c = { l__enumext_fake_item_indent_#2_dim },
          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 },
887
          list-offset .value_required:n = true,
888
          list-indent
                        .meta:n
                                 = { list-offset = ##1 },
          list-indent .value_required:n = true,
891
892
893 \clist_map_inline:nn
   {
      {enumext*}{vii}, {keyans*}{viii}
    }
    { \__enumext_tmp:nn #1 }
```

12.17.1 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.

```
898 \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
             }
911
         }
912
913
914 \cs_set_protected:Nn \__enumext_keyans_fake_item:
915
       \dim_compare:nNnT
916
         { \l__enumext_fake_item_indent_v_dim } > { \c_zero_dim }
917
           \tl_set:Ne \l__enumext_fake_item_indent_v_tl
```

__enumext_fake_item:
__enumext_keyans_fake_item:
__enumext_fake_item_vii:
__enumext_fake_item_viii:

```
\exp_not:N \mode_leave_vertical:
               \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_v_dim
        }
924
     }
925
926 \cs_set_protected:Nn \__enumext_fake_item_vii:
927
      \dim_compare:nNnT
928
         { \l__enumext_fake_item_indent_vii_dim } > { \c_zero_dim }
           \tl_set:Ne \l__enumext_fake_item_indent_vii_tl
             {
               \exp_not:N \mode_leave_vertical:
               \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_vii_dim
935
        }
936
937
  \cs_set_protected:Nn \__enumext_fake_item_viii:
938
939
      \dim_compare:nNnT
940
        { \l__enumext_fake_item_indent_viii_dim } > { \c_zero_dim }
         {
           \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
        }
     }
```

(End of definition for $__enumext_fake_item$: and others.)

12.18 Setting show-length key

show-length

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.)

12.19 Setting before, after and first keys

```
\keys_define:nn { enumext / #1 }
961
        {
          before .tl_set:c = { l__enumext_before_no_starred_key_#2_tl },
          before .value_required:n = true,
          before* .tl_set:c = { l__enumext_before_starred_key_#2_tl },
          before* .value_required:n = true,
966
                 .tl_set:c = { l__enumext_after_stop_list_#2_tl },
          after
                  .value_required:n = true,
          first
                  .tl_set:c = { l__enumext_after_list_args_#2_tl },
          first
                  .value_required:n = true,
971
        }
973 \clist_map_inline:Nn \c__enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

(End of definition for before and others.)

©2024 by Pablo González L 45/145

12.19.1 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 $\{\langle arg\ two \rangle\}$ of the list: $\{\langle code \rangle\}$ \list $\{\langle arg\ one \rangle\}$ $\{\langle arg\ two \rangle\}$.

```
974 \cs_new_protected:Nn \__enumext_before_args_exec:
975 {
976    \tl_use:c { l__enumext_before_starred_key_ \__enumext_level: _tl }
977 }
```

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 $\langle keys \rangle$: \list $\{\langle arg\ one \rangle\}$ $\{\langle arg\ two \rangle\}$

```
978 \cs_new_protected:Nn \__enumext_before_keys_exec:
979 {
980     \tl_use:c { l__enumext_before_no_starred_key_ \__enumext_level: _tl }
981 }
```

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

```
982 \cs_new_protected:Nn \__enumext_after_stop_list:
983 {
984     \tl_use:c { l__enumext_after_stop_list_ \__enumext_level: _tl }
985 }
```

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: \list{\arg one}\}{\arg two\}}{\arg two\}}\item.

```
9% \cs_new_protected:Nn \__enumext_after_args_exec:
9% {
9% \tl_use:c { l__enumext_after_list_args_ \__enumext_level: _tl }
9% }
```

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

12.19.2 Functions for before, after and first keys in keyans

__enumext_before_args_exec_v:
__enumext_before_keys_exec_v:
__enumext_after_stop_list_v:
__enumext_after_args_exec_v:

```
Same implementation as the one used in the enumext environment.
```

```
990 \cs_new_protected:Nn \__enumext_before_args_exec_v:
991 {
992    \tl_use:N \l__enumext_before_starred_key_v_tl
993 }
994 \cs_new_protected:Nn \__enumext_before_keys_exec_v:
995    {
996    \tl_use:N \l__enumext_before_no_starred_key_v_tl
997 }
998 \cs_new_protected:Nn \__enumext_after_stop_list_v:
999    {
1000    \tl_use:N \l__enumext_after_stop_list_v_tl
1001 }
1002 \cs_new_protected:Nn \__enumext_after_args_exec_v:
1003    {
1004    \tl_use:N \l__enumext_after_list_args_v_tl
1005 }
```

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

12.19.3 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:

(End of definition for $\ensuremath{\backslash}$ _enumext_before_args_exec_vii: and others.)

12.20 Setting keys for multicols and minipage

mini-env mini-sep columns-sep columns The default value of the columns-sep key is handled by the state of the boolean variable \l_enumext_-columns_sep_X_bool which is handled in the internal definition of the enumext and keyans environments. Define and set mini-env, mini-sep, columns-sep and columns keys for enumext, enumext*, keyans and keyans* environments.

```
\cs_set_protected:Npn \__enumext_tmp:nn #1 #2
1039
       \keys_define:nn { enumext / #1 }
1040
         {
1041
           mini-env
                      .dim_set:c = { l__enumext_minipage_right_#2_dim },
          mini-env
                     .value_required:n = true,
1043
          mini-sep
                     .dim_set:c = { l__enumext_minipage_hsep_#2_dim },
          mini-sep
                     .initial:n = 0.3333em,
          mini-sep
                      .value_required:n = true,
           columns-sep .dim_set:c = { l__enumext_columns_sep_#2_dim },
1047
           columns-sep .value_required:n = true,
1048
           columns
                      .int_set:c = { l__enumext_columns_#2_int },
1049
           columns
                       .initial:n = 1,
1050
           columns
                      .value_required:n = true,
1051
1052
1053
1054 \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 command \miniright is not available, so we will add the keys mini-right and mini-right* to implement support for minipage environment.

```
1055 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
1056
       \keys_define:nn { enumext / #1 }
1057
1058
           mini-right .tl_gset:c = { g__enumext_miniright_code_#2_tl },
                       .value_required:n = true,
           mini-right
           mini-right* .code:n
                                      \bool_gset_true:c { g__enumext_minipage_center_#2_bool }
                                      \keys_set:nn { enumext / #1 } { mini-right = {##1} }
                                    1.
           mini-right* .value_required:n = true,
1066
1067
1068 \clist_map_inline:nn { {enumext*}{vii}, {keyans*}{viii} } { \__enumext_tmp:nn #1 }
```

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

12.21 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.

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.

©2024 by Pablo González L

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

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.

12.21.1 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 $\langle vertical \ mode$

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*.

(End of definition for __enumext_multi_set_vskip:.)

__enumext_add_pre_parsep:

The function __enumext_add_pre_parsep: "adjusted" the value of \l__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*.

```
\cs_new_protected:Nn \__enumext_add_pre_parsep:
1082
    {
       \int_case:nn { \l__enumext_level_int }
1083
1084
         {
           { 2 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_i_skip } { \c_zero_skip }
                      \skip_add:Nn \l__enumext_multicols_above_ii_skip { \l__enumext_parsep_i_skip }
           { 3 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_ii_skip } { \c_zero_skip }
                      \skip_add:Nn \l__enumext_multicols_above_iii_skip { \l__enumext_parsep_ii_skip
           { 4 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_iii_skip } { \c_zero_skip }
                      \skip_add:Nn \l__enumext_multicols_above_iv_skip {    \l__enumext_parsep_iii_skip
                }
         }
```

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

©2024 by Pablo González L 48/145

__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_EX is in $\langle horizontal\ mode \rangle$ or $\langle vertical\ mode \rangle$.

(End of definition for $\ensuremath{\setminus} _$ enumext $_$ multi $_$ addvspace:.)

12.21.2 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:
       \skip_set:Nn \l__enumext_multicols_above_v_skip
           \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:
1134
       \__enumext_keyans_multi_set_vskip:
       \mode_if_vertical:T
           \skip_add:Nn \l__enumext_multicols_above_v_skip
               \skip use:N \l enumext partopsep v skip
1141
           \skip_add:Nn \l__enumext_multicols_below_v_skip
1142
               \skip_use:N \l__enumext_partopsep_v_skip
1144
       \par\nopagebreak
       \addvspace{ \l__enumext_multicols_above_v_skip }
```

 $(\textit{End of definition for } \verb|\|_enumext_keyans_multi_set_vskip: and \verb|\|_enumext_keyans_multi_addvspace:|)$

12.22 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.



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

©2024 by Pablo González L 49/145

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.

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 \(\lambda \text{vertical mode} \rangle \). Depending on these cases, small adjustments must be made using \vspace and \addvspace to obtain the "desired vertical spacing".

■ 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[14] package, again my attempts to find the correct values using \showoutput and \showboxdepth absolutely failed.

12.22.1 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 $\langle partopsep$ 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.

```
\skip_if_eq:nnTF
             { \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip } } { \c_zero_skip }
1156
               \skip_set:Nn \l__enumext_minipage_left_skip
                    -0.150\box_dp:N \strutbox
                 }
               \skip_set:Nn \l__enumext_minipage_right_skip
                   0.695\box_dp:N \strutbox
               \skip_set:Nn \l__enumext_minipage_after_skip
                    \box_dp:N \strutbox
                 }
               \__enumext_zero_parsep:
             }
               \skip_set:Nn \l__enumext_minipage_left_skip
                 {
                    \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
               \skip_set:Nn \l__enumext_minipage_right_skip
                 {
                   0.695\box_dp:N \strutbox
                 }
               \skip_set:Nn \l__enumext_minipage_after_skip
                 {
1182
                   1.85\box_dp:N \strutbox
                    + \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
1184
1185
             }
1186
         }
1187
```

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
1203
             }
1205
1206
               \skip_set:Nn \l__enumext_minipage_left_skip
1207
                 {
                   0.5875\box_dp:N \strutbox
                   - \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
                 7
               \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
                 {
                   0.325\box_dp:N \strutbox
                   + \skip_use:c { l__enumext_topsep_ \__enumext_level: _skip }
                 }
             }
         }
1224
```

(End of definition for __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.

```
\cs_new_protected:Nn \__enumext_zero_parsep:
1226
       \int_case:nn { \l__enumext_level_int }
         {
           { 2 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_i_skip } { \c_zero_skip }
                       \skip_add:Nn \l__enumext_minipage_after_skip { 2.15\box_dp:N \strutbox }
           { 3 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_ii_skip } { \c_zero_skip }
                       \skip_add:Nn \l__enumext_minipage_after_skip { 2.15\box_dp:N \strutbox }
           { 4 }{
                  \skip_if_eq:nnF { \l__enumext_parsep_iii_skip } { \c_zero_skip }
1243
                       \skip_add:Nn \l__enumext_minipage_after_skip { 2.15\box_dp:N \strutbox }
1244
1245
                }
         }
1247
     }
```

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

__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 \(\lambda \) horizontal mode \)

©2024 by Pablo González L

or $\langle vertical\ mode \rangle$. For the latter we will make some adjustments since the \partopsep parameter comes into play and this affects the $vertical\ spacing$.

```
\cs_new_protected:Nn \__enumext_mini_addvspace:
1250
       \__enumext_mini_set_vskip:
1251
       \mode_if_vertical:T
         {
           \skip_add:Nn \l__enumext_minipage_left_skip
                \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
             }
           \skip_add:Nn \l__enumext_minipage_after_skip
             {
                \skip_use:c { l__enumext_partopsep_ \__enumext_level: _skip }
             }
1261
         }
1262
       \par\nopagebreak
1263
       \addvspace { \l__enumext_minipage_left_skip }
1264
1265
```

(End of definition for __enumext_mini_addvspace:.)

12.22.2 Adjustment of vertical spaces for minipage in keyans

 $\verb|__enumext_keyans_mini_set_vskip:|$

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.

```
1266 \cs_new_protected:Nn \__enumext_keyans_mini_set_vskip:
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
       \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 }
1276
               \skip_set:Nn \l__enumext_minipage_after_skip { \box_dp:N \strutbox }
               \skip_if_eq:nnF { \l__enumext_parsep_i_skip } { \c_zero_skip }
                    \skip_add:Nn \l__enumext_minipage_after_skip { 2.15\box_dp:N \strutbox }
1280
                  }
               \skip_set:Nn \l__enumext_minipage_left_skip
                    \skip_use:N \l__enumext_topsep_v_skip
                 }
               \skip_set:Nn \l__enumext_minipage_right_skip
                    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
                  }
             3
         }
1297
1298
           \skip_if_eq:nnTF { \l__enumext_topsep_v_skip } { \c_zero_skip }
1299
1300
               \skip_set:Nn \l__enumext_minipage_left_skip
1301
                 {
1302
                    0.5\box_dp:N \strutbox
1303
                    + \l__enumext_partopsep_v_skip
1304
               \skip_set:Nn \l__enumext_minipage_right_skip
1307
                  {
1308
                    \l__enumext_partopsep_v_skip
1309
```

(End of definition for __enumext_keyans_mini_set_vskip:.)

__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 \(\lambda \text{horizontal mode} \rangle \text{ or } \sqrt{vertical mode} \rangle. \) 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.

```
\cs_new_protected:Nn \__enumext_keyans_mini_addvspace:
1329
         _enumext_keyans_mini_set_vskip:
       \mode_if_vertical:T
1331
           \skip add:Nn \l enumext minipage left skip
1334
                \l__enumext_partopsep_v_skip
1336
           \skip_add:Nn \l__enumext_minipage_after_skip
1338
                \l__enumext_partopsep_v_skip
       \par\nopagebreak
       \addvspace { \l__enumext_minipage_left_skip }
1343
1344
```

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

12.22.3 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:
1346
       \skip_zero_new:N \l__enumext_minipage_left_skip
1347
       \skip_gzero_new:N \g__enumext_minipage_right_skip
1348
       \skip_gzero_new:N \g__enumext_minipage_after_skip
1349
       \skip_if_eq:nnTF { \l__enumext_topsep_vii_skip } { \c_zero_skip }
         {
1351
           \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
1360
           \skip_gset:Nn \g__enumext_minipage_after_skip
1361
               0.325\box_dp:N \strutbox + \l__enumext_topsep_vii_skip
1363
```

```
}
     7
   \cs_new_protected:Nn \__enumext_mini_set_vskip_viii:
1368
       \skip_zero_new:N \l__enumext_minipage_after_skip
1369
       \skip_zero_new:N \l__enumext_minipage_left_skip
       \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
1374
               0.5\box_dp:N \strutbox
           \skip_set:Nn \l__enumext_minipage_right_skip
             {
                \l__enumext_partopsep_viii_skip
1381
           \skip_set:Nn \l__enumext_minipage_after_skip
1382
             {
1383
               1.6\box_dp:N \strutbox
         }
           \skip_set:Nn \l__enumext_minipage_left_skip
1389
               0.5875\box_dp:N \strutbox
1391
           \skip_set:Nn \l__enumext_minipage_right_skip
1392
1393
               \l__enumext_topsep_viii_skip
1394
           \skip_set:Nn \l__enumext_minipage_after_skip
             {
               0.325\box_dp:N \strutbox + \l__enumext_topsep_viii_skip
             }
          }
1401
```

(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 $\mathit{left side}$ when the mini-right key is active in the enumext* and keyans* environments.

Here we will NOT take into account whether TeX is in (horizontal mode) or (vertical mode), since \partopsep is equal to opt in both environments.

```
\cs_new_protected:Nn \__enumext_mini_addvspace_vii:
1403
       \__enumext_mini_set_vskip_vii:
1404
       \par\nopagebreak
1405
       \addvspace { \l__enumext_minipage_left_skip }
\cs_new_protected:Nn \__enumext_mini_addvspace_viii:
       \__enumext_mini_set_vskip_viii:
       \par\nopagebreak
1411
       \addvspace { \l__enumext_minipage_left_skip }
1412
1413
```

(End of definition for __enumext_mini_addvspace_vii: and __enumext_mini_addvspace_viii:.)

12.22.4 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 argument '*' inhibits the use of \centering command i.e. the usual MTx justification is maintained in the __enumext_mini_env* on the "right side".

First we will perform some checks to prevent the command from being executed outside the enumext \miniright environment or somewhere inappropriate then we will call the internal functions to execute it in the enumext and keyans environments.

```
\NewDocumentCommand \miniright { s }
1415
       \int_compare:nNnT { \l__enumext_keyans_pic_level_int } = { 1 }
1417
           \msg_error:nnn { enumext } { wrong-miniright-place }
1418
         }
       % outside
       \bool_lazy_and:nnT
1421
         { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
1422
         { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
           \msg_error:nnn { enumext } { wrong-miniright-place }
         }
       % starred env
1427
       \bool_if:NT \l__enumext_starred_bool
1428
         {
           \msg_error:nnn { enumext } { wrong-miniright-starred }
1430
1431
       \int_compare:nNnTF { \l__enumext_keyans_level_int } = { 1 }
1432
1433
            \__enumext_keyans_mini_right_cmd:n {#1}
         { \__enumext_mini_right_cmd:n {#1} }
1437
```

(End of definition for \miniright. This function is documented on page 10.)

__enumext_mini_right_cmd:n

The function __enumext_mini_right_cmd:n takes as argument the *starred* '*' 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
     {
1439
       \dim_compare:nNnTF
1440
         { \dim_use:c { l__enumext_minipage_right_ \__enumext_level: _dim } } > { \c_zero_dim }
1441
1442
            \__enumext_multicols_stop:
1443
           \end{__enumext_mini_env*}
1444
           \hfill
           \begin{__enumext_mini_env*}
             { \dim_use:c { l__enumext_minipage_right_ \__enumext_level: _dim } }
             \par\addvspace { \l__enumext_minipage_right_skip }
             \bool_if:nF {#1}
               {
1450
                  \centering
1451
1452
             \int_gzero:N \g__enumext_minipage_stat_int
1453
         }
1454
         { \msg_error:nnn { enumext } { wrong-miniright-use } }
1455
       % paranoia
       \RenewDocumentCommand \miniright { s }
1458
           \msg_error:nn { enumext } { many-miniright-used }
         }
1460
1461
```

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

__enumext_keyans_mini_right_cmd:n

The function __enumext_keyans_mini_right_cmd:n takes as argument the *starred* '*' 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.

```
1462 \cs_new_protected:Npn \__enumext_keyans_mini_right_cmd:n #1
1463 {
1464 \dim_compare:nNnTF { \l__enumext_minipage_right_v_dim } > { \c_zero_dim }
1465 {
1466 \__enumext_keyans_multicols_stop:
1467 \end{__enumext_mini_env*}
1468 \hfill
©2024 by Pablo González L
```

```
\begin{__enumext_mini_env*}{ \l__enumext_minipage_right_v_dim }
             \par\addvspace { \l__enumext_minipage_right_skip }
             \bool_if:nF {#1}
               {
                 \centering
             \int_gzero:N \g__enumext_minipage_stat_int
         { \msg_error:nnn { enumext } { wrong-miniright-use } }
1477
       \RenewDocumentCommand \miniright { s }
           \msg_error:nn { enumext } { many-miniright-used }
         }
1481
     }
1482
```

(End of definition for __enumext_keyans_mini_right_cmd:n.)

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.

Define above, above*, below and below* keys for enumext and keyans environments.

```
ahove
above*
        \tag{1483} \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
 below
                \keys_define:nn { enumext / #1 }
below*
        1485
                  {
        1486
                           .skip_set:c = { l__enumext_vspace_above_#2_skip },
                    above
        1487
                    above .value_required:n = true,
        1488
                    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,
                    below
                           .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,
        1496
        1497
                  }
        1498
        1499 \clist_map_inline:Nn \c_enumext_all_envs_clist { \__enumext_tmp:nn #1 }
```

12.23.1 Functions for above and below keys in enumext

(End of definition for above and others.)

_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:
1501
    {
       \skip_if_eq:nnF
1502
         { \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } } { \c_zero_skip }
1503
1504
           \bool_if:cTF { l__enumext_vspace_a_star_ \__enumext_level: _bool }
1506
               \vspace*{ \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } }
             }
               \vspace { \skip_use:c { l__enumext_vspace_above_ \__enumext_level: _skip } }
         }
```

(End of definition for __enumext_vspace_above:.)

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

```
\cs_new_protected:Nn \__enumext_vspace_below:
      \skip_if_eq:nnF
```

©2024 by Pablo González L 56 / 145

```
{ \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } } { \c_zero_skip }

{ \bool_if:cTF { l__enumext_vspace_b_star_ \__enumext_level: _bool }

{ \vspace*{ \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } }

}

**vspace* \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } }

**vspace { \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } }

**vspace { \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } }

**jsssymptote

**vspace { \skip_use:c { l__enumext_vspace_below_ \__enumext_level: _skip } }

**jsssymptote

**jsssymptote
```

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

12.23.2 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:.)

12.23.3 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:
1551
       \skip_if_eq:nnF { \l__enumext_vspace_above_vii_skip } { \c_zero_skip }
           \bool_if:NTF \l__enumext_vspace_a_star_vii_bool
               \vspace*{ \l__enumext_vspace_above_vii_skip }
1556
             { \vspace { \l__enumext_vspace_above_vii_skip } }
         }
\cs_new_protected:Nn \__enumext_vspace_above_viii:
1562
       \skip_if_eq:nnF { \l__enumext_vspace_above_viii_skip } { \c_zero_skip }
1563
1564
           \bool_if:NTF \l__enumext_vspace_a_star_viii_bool
1565
1566
               \vspace*{ \l__enumext_vspace_above_viii_skip }
             { \vspace { \l__enumext_vspace_above_viii_skip } }
         }
©2024 by Pablo González L
```

(End of definition for __enumext_vspace_above_vii: and __enumext_vspace_above_viii:.)

 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.

```
1572 \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
1576
               \vspace*{ \l__enumext_vspace_below_vii_skip }
             { \vspace { \l__enumext_vspace_below_vii_skip } }
1581
     }
1582
   \cs_new_protected:Nn \__enumext_vspace_below_viii:
1583
1584
       \skip_if_eq:nnF { \l__enumext_vspace_below_viii_skip } { \c_zero_skip }
1585
1586
           \bool_if:NTF \l__enumext_vspace_b_star_viii_bool
               \vspace*{ \l__enumext_vspace_below_viii_skip }
             { \vspace { \l__enumext_vspace_below_viii_skip } }
         }
1592
     }
1593
```

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

12.24 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

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

```
resume
            \cs_set_protected:Npn \__enumext_tmp:n #1
resume*
         1595
                 \keys_define:nn { enumext / #1 }
         1596
                   {
                     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:,
         1601
                     resume* .value_forbidden:n = true,
         1602
         1603
         1604
         1605 \clist_map_inline:nn { level-1, enumext* } { \__enumext_tmp:n {#1} }
```

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

12.24.1 Internal functions for series key

__enumext_filter_series:n __enumext_filter_series_key:n \ enumext filter series pair:nn

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, resume* and base-fix 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, start*, save-ans and save-key keys.

 $(End\ of\ definition\ for\ _enumext_filter_series:n,\ _enumext_filter_series_key:n,\ and\ _enumext_filter_series_pair:nn.)$

__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 \(\lambda \text{keys} \rangle \). This function is passed to the function __enumext_parse_keys:n in the enumext environment definition (\(\subseteq 12.38 \)) and to the function __enumext_parse_keys_vii:n in the enumext* environment definition (\(\subseteq 12.43 \)).

```
1632 \cs_new_protected:Npn \__enumext_parse_series:n #1
1633
    {
      \str_if_empty:NTF \l__enumext_series_str
1635
          \bool_if:NF \l__enumext_resume_active_bool
1637
                _enumext_resume_last:n {#1}
        }
          \tl_gclear_new:c { g__enumext_series_ \l__enumext_series_str _tl }
          \tl_gset:ce { g__enumext_series_ \l__enumext_series_str _tl }
            \int_if_exist:cF { g__enumext_series_ \l__enumext_series_str _int }
              \int_new:c { g__enumext_series_ \l__enumext_series_str _int }
1647
1648
        }
1649
```

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 $_$ enumext_standar_series_tl for the enumext environment and in the variable $_$ 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.

```
\cs_new_protected:Npn \__enumext_resume_last:n #1
     {
1652
       \bool_if:NT \l__enumext_standar_first_bool
1653
1654
           \tl_gclear:N \g__enumext_standar_series_tl
           \tl_gset:Ne \g__enumext_standar_series_tl { \__enumext_filter_series:n {#1} }
       \bool_if:NT \l__enumext_starred_first_bool
         {
           \tl_gclear:N \g__enumext_starred_series_tl
1660
           \tl_gset:Ne \g__enumext_starred_series_tl { \__enumext_filter_series:n {#1} }
1661
         }
1662
1663
```

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

12.24.2 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 (\§12.38) and the enumext* environment definition (\§12.43).

```
\cs_new_protected:Nn \__enumext_resume_save_counter:
1665
     {
       \bool_if:NT \g__enumext_standar_bool
1666
1667
           \tl_if_empty:NF \l__enumext_series_str
1668
             {
1660
                \int_gset_eq:cN
1670
                  { g__enumext_series_ \l__enumext_series_str _int } \value{enumXi}
1671
           \tl_if_empty:NTF \l__enumext_resume_name_tl
             {
                \str_if_empty:NT \l__enumext_series_str
                  {
1676
                    \int_gset_eq:NN \g__enumext_resume_int \value{enumXi}
1677
                  }
1678
             }
                \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
                  {
                    \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 }
1688
                \int_gset_eq:cN
1680
                  { g__enumext_resume_ \l__enumext_store_name_tl _int } \value{enumXi}
         }
       \bool_if:NT \g__enumext_starred_bool
         {
           \tl_if_empty:NF \l__enumext_series_str
             {
                \int_gset_eq:cN
1607
                  { g__enumext_series_ \l__enumext_series_str _int } \value{enumXvii}
1698
1699
            \tl_if_empty:NTF \l__enumext_resume_name_tl
1700
             {
1701
                \str_if_empty:NT \l__enumext_series_str
                  {
                    \int_gset_eq:NN \g__enumext_resume_vii_int \value{enumXvii}
             }
                \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
                  {
                    \int_gset_eq:cN
                      { g__enumext_series_ \l__enumext_resume_name_tl _int } \value{enumXvii}
             }
           \int_if_exist:cT { g__enumext_resume_ \l__enumext_store_name_tl _int }
             {
                \int_gset_eq:cN
                   \{ \  \, {\tt g\_enumext\_resume\_} \  \, {\tt l\_enumext\_store\_name\_tl\_int} \  \, {\tt value\{enumXvii\}} \\
             3
         }
1720
```

12.24.3 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.

```
\cs_new_protected:Npn \__enumext_resume_series:n #1
       \tl_if_empty:nTF {#1}
              _enumext_resume_counter:n { }
         }
         {
           \tl_if_exist:cTF { g__enumext_series_ \tl_to_str:n {#1} _tl }
1728
                  _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* }
1738
                      { g__enumext_series_ \tl_to_str:n {#1} _tl }
1740
             }
1741
1742
               \bool_if:NT \g__enumext_standar_bool
                    \msg_error:nnn { enumext } { unknown-series } {#1}
                 }
               \bool_if:NT \g__enumext_starred_bool
                  {
1748
                    \msg_error:nnn { enumext } { unknown-series } {#1}
1749
                  }
         }
1752
     }
```

__enumext_resume_counter:n __enumext_resume_counter: _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.

```
1754 \cs_new_protected:Npn \__enumext_resume_counter:n #1
1755 {
1756     \bool_set_true:N \l__enumext_resume_active_bool
1757     \tl_set:Nn \l__enumext_resume_name_tl {#1}
1758     \tl_if_empty:NTF \l__enumext_resume_name_tl
1759     {
1760          \__enumext_resume_counter:
1761     }
1762     {
1763          \__enumext_resume_counter_series:
1764     }
1765     \__enumext_resume_counter_save_ans:
1766 }
```

(End of definition for __enumext_resume_series:n.)

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.

```
1767 \cs_new_protected:Nn \__enumext_resume_counter:
1768 {
1769 \bool_if:NT \g__enumext_standar_bool
©2024 by Pablo González L
```

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:
1781
     {
       \bool_if:NT \g__enumext_standar_bool
1782
1783
           \int_set:Nn \l__enumext_start_i_int
1784
1785
                \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
1786
1787
1788
       \bool_if:NT \g__enumext_starred_bool
           \int_set:Nn \l__enumext_start_vii_int
             {
                \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
1794
         }
1795
1796
```

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 }
1801
         {
1802
           \int set:Nn \l enumext start i int
1803
1804
             {
                \int_use:c { g__enumext_resume_ \l__enumext_store_name_tl _int } + 1
1805
       \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
1812
1813
                \int_use:c { g__enumext_resume_ \l__enumext_store_name_tl _int } + 1
1814
1815
         }
1816
1817
```

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

12.24.4 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.

©2024 by Pablo González L

```
\keys_set:nV { enumext / level-1 } \g__enumext_standar_series_tl
1826
         }
1827
       \bool_if:NT \g__enumext_starred_bool
1828
1829
            \tl_if_empty:NF \g__enumext_starred_series_tl
1820
1831
                 \__enumext_resume_counter:n { }
1832
                \keys_set:nV { enumext / enumext* } \g__enumext_starred_series_tl
1833
         }
1836
```

(End of definition for __enumext_resume_starred:.)

12.25 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.

12.25.1 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.)

12.25.2 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:.

```
1846 \cs_new_protected:Nn \__enumext_start_save_ans_msg:
1847 {
1848 \msg_term:nnVV { enumext } { save-ans-log }
1849 \g__enumext_envir_name_tl \l__enumext_store_name_tl
1850 }
1851 \cs_new_protected:Nn \__enumext_stop_save_ans_msg:
1852 {
1853 \msg_term:nnVV { enumext } { save-ans-log-hook }
1854 \g__enumext_envir_name_tl \g__enumext_store_name_tl
1855 }
1860 {enumext_start_save_ans_msg: and \__enumext_stop_save_ans_msg:)
```

__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.

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, copy $\{\langle store\ name \rangle\}$ into the global variable \g__enumext_store_name_tl and execute the function __enumext_anskey_env_make: V creating the environment anskey* (\\$12.30). The $\langle prop\ list \rangle$ \g__enumext_series_ $\langle store\ name \rangle$ _prop and the $\langle sequence \rangle$ \g_-enumext_series_ $\langle store\ name \rangle$ _int used by the keys resume and resume*.

```
1876 \cs_new_protected:Nn \__enumext_storing_exec:
1877
       \bool_set_true:N \l__enumext_store_active_bool
1878
       \bool_set_true:N \l__enumext_check_answers_bool
1879
       \tl_gset:NV \g__enumext_store_name_tl \l__enumext_store_name_tl
       \__enumext_anskey_env_make:V \l__enumext_store_name_tl
       \prop_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _prop }
           \msg_log:nnV { enumext } { store-prop } \l__enumext_store_name_tl
           \prop_new:c { g__enumext_ \l__enumext_store_name_tl _prop }
         }
       \seq_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _seq }
1887
1888
         {
           \msg_log:nnV { enumext } { store-seq } \l__enumext_store_name_tl
1889
           \seq_new:c { g__enumext_ \l__enumext_store_name_tl _seq }
1890
1891
       \int_if_exist:cF { g__enumext_resume_ \l__enumext_store_name_tl _int }
           \msg_log:nnV { enumext } { store-int } \l__enumext_store_name_tl
           \int_new:c { g__enumext_resume_ \l__enumext_store_name_tl _int }
         }
1896
1897
```

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

12.25.3 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 \item and \item* in the environment \g__enumext_-item_number_int must match the integer variable \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.

12.25.4 Setting check-ans and no-store keys

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

```
\cs_set_protected:Npn \__enumext_tmp:n #1
1899
       \keys_define:nn { enumext / #1 }
1900
         {
1901
           check-ans .bool_set:N = \l__enumext_check_ans_key_bool,
1902
           check-ans .initial:n = false,
1903
           check-ans .value_required:n = true,
           no-store .code:n = {
                                   \bool_set_false:N \l__enumext_check_answers_bool
                                  \bool_set_false:N \l__enumext_check_ans_key_bool
                                },
                    .value_forbidden:n = true,
           no-store
         }
1911
   \clist_map_inline:nn
1912
1913
       level-1, level-2, level-3, level-4, enumext*
1915
     { \__enumext_tmp:n {#1} }
```

(End of definition for check-ans and no-store.)

12.25.5 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 and set \l__enumext_item_number_bool to "false".

```
1927 \cs_new_protected:Nn \__enumext_check_ans_level:
    {
1928
       \int_case:nn { \l__enumext_level_int }
1929
           { 1 }{
1931
                  \bool_lazy_all:nT
1932
                    {
                       { \bool_if_p:N \g__enumext_starred_bool }
1934
                       { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
                    }
                       \int_gdecr:N \g__enumext_item_number_int
                       \bool_set_false:N \l__enumext_item_number_bool
                }
           { 2 }{
                   \int_gdecr:N \g__enumext_item_number_int
                  \bool_set_false:N \l__enumext_item_number_bool
           { 3 }{
                   \int_gdecr:N \g__enumext_item_number_int
                  \bool_set_false:N \l__enumext_item_number_bool
```

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.

(End of definition for __enumext_check_ans_active: and __enumext_check_ans_level:.)

__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.

```
1970 \cs_new_protected:Nn \__enumext_check_ans_key_hook:
       \verb|\bool_lazy_and:nnT|\\
         { \bool_if_p:N \l__enumext_check_ans_key_bool }
1973
         { \bool_if_p:N \g__enumext_standar_bool }
1974
1975
           \bool_gset_true:N \g__enumext_check_ans_key_bool
1977
       \bool_lazy_and:nnT
         { \bool_if_p:N \l__enumext_check_ans_key_bool }
           \bool_if_p:N \g__enumext_starred_bool }
            \bool_gset_true:N \g__enumext_check_ans_key_bool
         }
     }
1084
```

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

__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:.

```
1985 \cs_new_protected:Nn \__enumext_item_answer_diff:
1986 {
1987 \int_gset:Nn \g__enumext_item_answer_diff_int
1988 {
1989 \int_sign:n { \g__enumext_item_number_int - \g__enumext_item_anskey_int }
1990 }
1991 }
```

(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:.

66 / 145

```
1 }{ \__enumext_check_ans_msg_greater: }
    }
2001 \cs_new_protected:Nn \__enumext_check_ans_msg_less:
       \msg_warning:nneee { enumext } { item-less-answer } { \g_enumext_store_name_tl }
2003
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
    }
   \cs_new_protected:Nn \__enumext_check_ans_msg_same_ok:
       \msg_term:nneee { enumext } { items-same-answer } { \g_enumext_store_name_tl }
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2010
  \cs_new_protected:Nn \__enumext_check_ans_msg_greater:
2011
2012
       \msg_warning:nneee { enumext } { item-greater-answer } { \g__enumext_store_name_tl }
2013
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2014
2015
```

__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 $_$ _enumext_check_ans_key_bool is "false" and write in the log the appropriate message according to the value of $_$ _enumext_item_answer_diff_int set by the function $_$ _enumext_item_answer_diff:.

```
2016 \cs_new_protected:Nn \__enumext_check_ans_log:
    {
2017
       \int_case:nn { \g__enumext_item_answer_diff_int }
2018
2019
         {
           { -1 }{ \__enumext_check_ans_log_msg_less:
             0 }{ \__enumext_check_ans_log_msg_same_ok: }
2021
             1 }{ \__enumext_check_ans_log_msg_greater: }
2022
   \cs_new_protected:Nn \__enumext_check_ans_log_msg_less:
       \msg_log:nneee { enumext } { item-less-answer } { \g__enumext_store_name_tl }
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2028
2030 \cs new protected:Nn \ enumext check ans log msg same ok:
2031
       \msg_log:nneee { enumext } { items-same-answer } { \g__enumext_store_name_tl }
2032
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
    }
2035 \cs_new_protected:Nn \__enumext_check_ans_log_msg_greater:
2036
       \msg_log:nneee { enumext } { item-greater-answer } { \g__enumext_store_name_tl }
         { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
```

(End of definition for $\label{lem:lem:log:and}$ and others.)

 $(\mathit{End}\ of\ definition\ for\ \verb|_enumext_check_ans_show:\ and\ others.)$

12.25.6 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.

```
\tl_clear:N \l__enumext_check_start_line_env_tl
2056 }
(End of definition for \__enumext_check_starred_cmd:n.)
```

12.26 Keys and functions associated with storage

```
We add the keys wrap-ans, wrap-opt, save-sep, mark-ans, mark-pos, show-ans, show-pos, mark-
wrap-ans
          ref and save-ref related to the "storage system" and internal mechanism of "label and ref" only at the
save-sep first level of enumext and enumext*.
mark-ans
         2057 \cs_set_protected:Npn \__enumext_tmp:n #1
mark-pos 2058
show-ans 2059
                 \keys_define:nn { enumext / #1 }
mark-ref 2060
                                .cs_set_protected:Np = \__enumext_anskey_wrapper:n ##1,
                     wrap-ans
save-ref 2061
                                .initial:n =
                    wrap-ans
                                 {
                                    2064
                                  },
          2065
                     wrap-ans
                                .value_required:n = true,
                     wrap-opt
                                .cs_set_protected:Np = \__enumext_keyans_wrapper_opt:n ##1,
          2067
                     wrap-opt
                                .initial:n = [{##1}],
                     wrap-opt
                                .value_required:n = true,
                     save-sep
                                .tl_set:N = \l__enumext_store_keyans_item_opt_sep_tl,
                     save-sep
                                .initial:n = {, ~ },
                     save-sep
                                .value_required:n = true,
                               .tl_set:N = \l__enumext_mark_answer_sym_tl,
                    mark-ans
                               .initial:n = \textasteriskcentered,
                    mark-ans
          2074
                               .value_required:n = true,
                    mark-ans
                               .choice:,
                    mark-pos
          2076
                                       .code:n = \str_set:Nn \l__enumext_mark_position_str { l },
                    mark-pos / left
          2077
                                      .code:n = \str_set:Nn \l__enumext_mark_position_str { r },
                     mark-pos / right
          2078
                     mark-pos / unknown .code:n =
                                        \msg_error:nneee { enumext } { unknown-choice }
                                          { mark-pos } { left, ~ right } { \exp_not:n {##1} },
                     mark-pos
                                .initial:n = right,
                     mark-pos
                                .value_required:n = true,
                     show-ans
                               .bool_set:N = \l__enumext_show_answer_bool,
          2084
                     show-ans
                                .initial:n = false,
          2085
                     show-ans
                                .value_required:n = true,
          2086
                     show-pos
                               .bool_set:N = \l__enumext_show_position_bool,
          2087
                               .initial:n = false,
                     show-pos
          2088
                                .value_required:n = true,
                     show-pos
                     mark-ref
                                .tl_set:N = \l__enumext_mark_ref_sym_tl,
                    mark-ref
                               .initial:n = \textasteriskcentered,
                    mark-ref
                               .value_required:n = true,
                    save-ref
                               .bool_set:N = \l__enumext_store_ref_key_bool,
                               .initial:n = false,
                     save-ref
                     save-ref
                               .value required:n = true,
          2096
          2098 \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
          2099 \cs_set_protected:Npn \__enumext_tmp:n #1
                 \keys_define:nn { enumext / #1 }
                  {
                    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 },
          2105
                     mark-pos .initial:n = right,
          2106
                     mark-pos .value_required:n = true,
          2107
                     show-ans .bool_set:N = \l__enumext_show_answer_bool,
                     show-ans .initial:n = false,
                     show-ans .value_required:n = true,
                     show-pos .bool_set:N = \l__enumext_show_position_bool,
                     show-pos .initial:n = false,
                     show-pos .value_required:n = true,
          ©2024 by Pablo González L
```

```
2114      }
2115      }
2116 \clist_map_inline:nn { keyans, keyans* } { \__enumext_tmp:n {#1} }
```

(End of definition for mark-pos, show-ans, and show-pos.)

12.26.1 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 $\langle keys \rangle$.

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

12.26.2 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*.

```
2149 \cs_new_protected:Npn \__enumext_parse_save_key:n #1
2150 {
2151 \bool_set_true:c { l__enumext_store_save_key_ \__enumext_level: _bool }
©2024 by Pablo González L
```

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

12.26.3 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*, no-store and base-fix keys.

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

```
2180 \cs_new:Npn \__enumext_filter_save_key_pair:nn #1#2
2181
       \str_case:nnF {#1}
2182
         {
           { series
                      } {} { resume
                                         } {} { save-ans } {} { save-ref
           { save-key } {} { check-ans } {} { show-ans } {} { show-pos
                                                                             } {}
           { wrap-ans } {} { mark-ans } {} { wrap-opt } {} { save-sep
                                                                             } {}
           { mark-ref } {} { mini-env } {} { mini-sep } {} { mini-right } {}
           { mini-right* } {}
2188
2189
         { , { \exp_not:n {#1} } = { \exp_not:n {#2} } }
2190
2191
```

(End of definition for $_$ enumext_filter_save_key:n, $_$ enumext_filter_save_key_key:n, and $_$ enumext_filter_save key pair:nn.)

12.26.4 Function for storing content in prop list

__enumext_store_addto_prop:n
__enumext_store_addto_prop:V

The function __enumext_store_addto_prop:n stores the content in $\langle prop \ list \rangle$ defined by save-ans key. The "stored content" is retrieved by means of the \getkeyans command.

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.

70 / 145

©2024 by Pablo González L

```
2199  }
2200 \cs_generate_variant:Nn \__enumext_store_addto_prop:n { V, e }

(End of definition for \__enumext_store_addto_prop:n.)
```

12.26.5 Function for storing content in sequence

__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 with the same structure in which the command was executed.

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

```
2201 \cs_new_protected:Npn \__enumext_store_addto_seq:n #1
2202 {
2203    \seq_gput_right:cn { g__enumext_ \l__enumext_store_name_tl _seq } { #1 }
2204    }
2205 \cs_generate_variant:Nn \__enumext_store_addto_seq:n { v, V, e }
```

(End of definition for $\ensuremath{\setminus}$ enumext_store_addto_seq:n.)

12.26.6 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.

```
2206 \cs_new_protected:Nn \__enumext_store_level_open:
2207
       \bool_if:NT \l__enumext_check_answers_bool
2208
           \tl_if_empty:cTF { l__enumext_store_save_key_ \__enumext_level: _tl }
2211
                  _enumext_store_addto_seq:n
                   \item \begin{enumext}
             }
               \tl_put_left:cn { l__enumext_store_save_key_ \__enumext_level: _tl }
                   \item \begin{enumext} [
                 }
               \tl_put_right:cn { l__enumext_store_save_key_ \__enumext_level: _tl }
                 }
                 _enumext_store_addto_seq:v { l__enumext_store_save_key_ \__enumext_level: _tl }
         }
2228
2229
   \cs_new_protected:Nn \__enumext_store_level_close:
2230
       \bool_if:NT \l__enumext_check_answers_bool
             _enumext_store_addto_seq:n { \end{enumext} }
```

(End of definition for __enumext_store_level_open: and __enumext_store_level_close:.)

__enumext_store_level_open_vii: __enumext_store_level_close_vii:

The memorization structure of the list is handled by the functions __enumext_store_level_open_vii: and __enumext_store_level_close_vii: which are executed in the enumext* environment.

©2024 by Pablo González L

```
\tl_put_left:Nn \l__enumext_store_save_key_vii_tl
                    \item \begin{enumext*}[
                  7
                \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:
2261
2262
       \bool_if:NT \l__enumext_check_answers_bool
2263
         {
2264
             _enumext_store_addto_seq:n { \end{enumext*} }
2265
2266
2267
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_enumext_store_level_open_vii:\ and\ \verb|_enumext_store_level_close_vii:.)$

12.26.7 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:

12.27 The internal label and ref

(End of definition for $\ensuremath{\setminus} _$ enumext $_$ print $_$ keyans $_$ box:NN.)

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.

__enumext_store_internal_ref:

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

cs_set_protected:Npn \__enumext_tmp:n ##1

cs_set_protected:Npn \__enumext_tmp:n ##1

cs_set_protected:Npn \__enumext_tmp:n ##1

cs_set_eq:cc { l__enumext_label_copy_##1_tl } { l__enumext_label_##1_tl }

cs_set_eq:cc { l__enumext_label_copy_##1_tl } { . }

cs_set:Npn \__enumext_label_copy_##1_tl }

cs_set:Npn \__enumext_tmp:n ##1

cs_set:Npn \__enumext_tmp:n ##1

cs_set:Npn \__enumext_label_copy_ \int_to_roman:n {##1} _tl }

cs_set:Npn \_enumext_label_copy_ \int_to_roman:n {##1} _tl }

cs_set:Npn \_enumext
```

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

```
bool_lazy_all:nT

2295 {

©2024 by Pablo González L
```

```
{ \bool_if_p:N \g__enumext_starred_bool }
                                                                \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
                                             }
2298
                                             {
2299
                                                         \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2300
                                                                  { \tl_use:N \l__enumext_label_copy_vii_tl }
2301
                                   \bool_lazy_all:nT
2303
                                              {
                                                        { \bool_not_p:n { \g__enumext_standar_bool } }
                                                        { \bool_if_p:N \l__enumext_standar_bool }
                                                        { \left\{ \begin{array}{c} {\cluster} \\ {\clus
                                             }
                                             {
                                                        \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 or anskey* is running alone in it or if it is running in a nested enumext* environment within the starting environment.

```
\bool lazy all:nT
2216
         {
           { \bool_if_p:N \g__enumext_standar_bool }
2318
           { \int_compare_p:nNn { \l__enumext_level_int } > { 0 } }
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
               \tl_use:N \l__enumext_label_copy_i_tl
               \int_step_function:nnN { 2 } { \l__enumext_level_int } \__enumext_tmp:n
2226
         }
2328
       \cs_set:Npn \__enumext_tmp:n ##1
         { \tl_use:c { l__enumext_label_copy_ \int_to_roman:n {##1} _tl } . }
       \bool_lazy_all:nT
           { \bool_if_p:N \g__enumext_standar_bool }
           { \bool_if_p:N \l__enumext_starred_bool }
           { \int_compare_p:nNn { \l__enumext_level_int } > { 0 } }
         }
         {
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2338
2339
               \int_step_function:nnN { 1 } { \l__enumext_level_int } \__enumext_tmp:n
2340
               \tl_use:N \l__enumext_label_copy_vii_tl
2341
```

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

```
2344 \tl_put_right:Ne \l__enumext_newlabel_arg_one_tl
2345 {
2346 \l__enumext_store_name_tl \c_colon_str
2347 \int_eval:n { \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } }
2348 }
```

Now execute the function __enumext_newlabel:nn and save the result in the variable \l__enumext_-write_aux_file_tl and finally we write in the .aux file.

(End of definition for __enumext_store_internal_ref:.)

12.28 Common functions for \anskey and anskey* environment

__enumext_store_anskey_code:n

The internal function __enumext_store_anskey_code:n first we pass the $\langle argument \rangle$ 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" $\langle argument \rangle$.

```
2357 \cs_new_protected:Npn \__enumext_store_anskey_code:n #1
2358 {
2359 \int_gincr:N \g__enumext_item_anskey_int
2360 \__enumext_store_addto_prop:n {#1}
2361 \bool_if:NT \l__enumext_store_ref_key_bool
2362 {
2363 \__enumext_store_internal_ref:
2364 }
2365 \__enumext_anskey_show_wrap_left:n { #1 }
```

Now we start processing the $\lceil \langle key = val \rangle \rceil$ 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 the $\langle keys \rangle$, if the break-col key is present and the command is running under enumext (not in enumext*) we will add \columnbreak and then \item.

```
\tl_clear:N \l__enumext_store_anskey_arg_tl

\text{bool_lazy_and:nnT}

\text{\bool_if_p:N \l__enumext_store_columns_break_bool} \\

\text{\bool_not_p:n { \l__enumext_starred_bool} } \\

\text{\text{\text{tl_put_left:Nn \l__enumext_store_anskey_arg_tl} { \columnbreak} } \\

\text{\text{\text{tl_put_left:Nn \l__enumext_store_anskey_arg_tl} { \columnbreak} } \\

\text{\text{\text{tl_put_right:Nn \l_enumext_store_anskey_arg_tl} { \columnbreak} } \\

\text{\text{\text{tl_put_right:Nn \l_enumext_store_anskey_arg_tl} } \\

\text{\text{\text{tl_put_right:Nn \l_enumext_store_anskey_arg_tl} } \\

\text{\text{\text{\text{tl_put_right:Nn \l_enumext_store_anskey_arg_tl} } \\

\text{\text{\text{\text{\text{\text{tl_put_right:Nn \l_enumext_store_anskey_arg_tl} } } \\

\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\text{\t
```

If the item-join key is present and the command is running under enumext* we will add $(\langle number \rangle)$ to $\l_enumext_store_anskey_arg_tl$.

And now we will review the keys item-star, item-sym* and item-pos* and pass them to \l_-enumext_store_anskey_arg_tl along with the $\langle argument \rangle$ for \anskey or $\langle body \rangle$ for anskey*.

```
\bool_if:NTF \l__enumext_store_item_star_bool
         {
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl { * }
           \tl_if_empty:NF \l__enumext_store_item_symbol_tl
2386
             {
               \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2388
                 {
                   [ \exp_not:V \l__enumext_store_item_symbol_tl ]
                 }
             }
           \dim_compare:nT
             {
               \l__enumext_store_item_symbol_sep_dim != \c_zero_dim
             }
             {
               \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
                 {
                     \exp_not:V \l__enumext_store_item_symbol_sep_dim ]
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#1}
         }
         {
           \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#1}
```

Finally we check if the save-ref key are active along with the hyperref package load, if both conditions are met, it will create the hyperlink with symbol set by mark-ref key and then store in sequence.

```
\bool_lazy_and:nnT
```

__enumext_anskey_show_wrap_arg:n

The function __enumext_anskey_show_wrap_arg:n "wraps" the $\langle argument \rangle$ passed to \anskey and the $\langle body \rangle$ for anskey* when using the wrap-anskey.

```
\cs_new_protected:Npn \__enumext_anskey_show_wrap_arg:n #1
    {
       \bool_if:NTF \l__enumext_starred_bool
              _enumext_print_keyans_box:NN \l__enumext_labelwidth_vii_dim \l__enumext_labelsep_vii_d
         }
         {
             _enumext_print_keyans_box:cc
2428
             { l__enumext_labelwidth_ \__enumext_level: _dim }
2429
             { l__enumext_labelsep_ \__enumext_level: _dim }
2430
2431
       \__enumext_anskey_wrapper:n { #1 }
2432
2433
```

(End of definition for __enumext_anskey_show_wrap_arg:n.)

(End of definition for __enumext_store_anskey_code:n.)

__enumext_anskey_show_wrap_left:n

The function __enumext_anskey_show_wrap_left:n will show the "mark" defined by the mark-ans 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 and the $\langle body \rangle$ in anskey* on the right side when using the show-ans key.

```
2434 \cs_new_protected:Npn \__enumext_anskey_show_wrap_left:n #1
       \bool_if:NT \l__enumext_show_answer_bool
2437
         {
             _enumext_anskey_show_wrap_arg:n { #1 }
2438
       \bool_if:NT \l__enumext_show_position_bool
         {
2441
           \tl_set:Ne \l__enumext_mark_answer_sym_tl
2442
                \group_begin:
                \exp_not:N \normalfont
                \exp_not:N \footnotesize [ \int_eval:n
                    \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
2448
                 }
                  ٦
                \group_end:
2451
2452
             _enumext_anskey_show_wrap_arg:n { #1 }
         }
```

(End of definition for __enumext_anskey_show_wrap_left:n.)

12.29 The command \anskey

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 similar behaviour to that of \item in the enumext and enumext* environments executed as follows \anskey[$\langle key = val \rangle$] { $\langle content \rangle$ }.

©2024 by Pablo González L 75/145

__enumext_anskey_unknown:n
__enumext_anskey_unknown:nn

First we'll add the keys break-col, item-join, item-star, item-sym* and item-pos*.

```
2456 \keys_define:nn { enumext / anskey }
2457
      break-col .bool_set:N = \l__enumext_store_columns_break_bool,
2458
       break-col .default:n = true,
2459
       break-col .value_forbidden:n = true,
2460
       item-join .int_set:N = \l__enumext_store_item_join_int,
2461
       item-join .value_required:n = true,
       item-star .bool_set:N = \l__enumext_store_item_star_bool,
       item-star .default:n = true,
       item-star .value_forbidden:n = true,
       item-sym* .tl_set:N = \l__enumext_store_item_symbol_tl,
       item-sym* .value_required:n = true,
       item-pos* .dim_set:N = \l__enumext_store_item_symbol_sep_dim,
       item-pos* .value_required:n = true,
                            = { \__enumext_anskey_unknown:n {#1} },
       unknown .code:n
2471
```

The $\langle keys \rangle$ are stored in \l_keys_key_str and the value (if any) is passed as an argument to the function _enumext_anskey_unknown:n.

```
2472 \cs_new_protected:Npn \__enumext_anskey_unknown:n #1
2473
       \exp_args:NV \__enumext_anskey_unknown:nn \l_keys_key_str {#1}
2474
2475
   \cs_new_protected:Npn \__enumext_anskey_unknown:nn #1 #2
2476
2477
       \tl_if_blank:nTF {#2}
2479
           \msg_error:nnn { enumext } { anskey-cmd-key-unknown } {#1}
2480
         }
         {
           \msg_error:nnnn { enumext } { anskey-cmd-key-value-unknown } {#1} {#2}
         }
2484
     }
2485
```

 $(\textit{End of definition for } \verb|_=enumext_anskey_unknown:n and \verb|_=enumext_anskey_unknown:nn.|)$

The \anskey command will only be present when using the save-ans key in enumext and enumext* environments, otherwise it will return an error.

\anskey

We will first call the function __enumext_anskey_safe_outer: to be sure where we execute the command, then we will check the state of the variable \l__enumext_check_answers_bool set by the key no-store, if is true we will increment \g__enumext_item_anskey_int for the internal "check answer" system and execute the function __enumext_anskey_safe_inner:n to ensure that the command is not nested and that the argument is not empty, finally search the $[\langle key = val \rangle]$ and call the function __enumext_store_anskey_code:n.

```
2486 \NewDocumentCommand \anskey { o +m }
2487
       \__enumext_anskey_safe_outer:
2488
       \group_begin:
2489
         \bool_if:NT \l__enumext_check_answers_bool
2490
2491
              \tl_if_novalue:nF {#1}
                  \keys_set:nn { enumext / anskey } {#1}
              \tl_if_blank:nTF {#2}
               {
                  \msg_error:nn { enumext } { anskey-empty-arg }
                {
                  \__enumext_anskey_safe_inner:
2501
                  \__enumext_store_anskey_code:n {#2}
                }
           }
       \group_end:
     }
```

(End of definition for \anskey. This function is documented on page 12.)

©2024 by Pablo González L

76 / 145

12.29.1 Internal functions for the command

__enumext_anskey_safe_outer:
__enumext_anskey_safe_inner:

The __enumext_store_anskey_safe_outer: function will return the appropriate messages when the command is executed outside the environment in which the save-ans key was activated.

The __enumext_anskey_safe_inner: function will first check if the command is nested, if preceded by a not numbered \item or if it is in *math mode* returning the appropriate messages.

```
css_new_protected:Nn \__enumext_anskey_safe_inner:
{
css_new_protected:Nn \__enumext_anskey_level_int
css_new_protected:Nn \_enumext_anskey_level_int
css_new_protected:Nn \_enumext_anskey_level_i
```

 $(\mathit{End}\ of\ definition\ for\ \verb|_=enumext_anskey_safe_outer:\ and\ \verb|_=enumext_anskey_safe_inner:|)$

12.30 The environment anskey*

Managing *verbatim content* in an environment is quite complicated, I learned that when creating the **scontents** package, so to be able to have support at this point it is best to play a little with the internal code of **scontents** and *hooks*. Some considerations I should have here before implementing this:

- If some package, class or user has defined the environment with the same name somewhere in the document it would be a problem, you would not know what argument has been passed to store-env, if you are using the key print-env or the write-out key, sure, I can detect and modify it within the enumext and enumext* environments, but it would look strange not to have some keys available when running within these environments.
- A better (perhaps a bit paranoid) option is to define it within the environment in which the save-ans key is executed. and have it available only when that key is executed, here I would have absolute control of the \(\lambda \text{keys} \rangle \) and I make sure that write-out is not used, then using hooks after I undefine it and using hook before I check if it has been created by any package, class or user and I return a error, then the user will have to see how to solve the problem.

__enumext_undefine_anskey_env:

The function $_$ _enumext_undefine_anskey_env: will undefine the environment anskey* and will be passed to the function $_$ _enumext_execute_after_env: ($\S12.31$) which is executed after the environment in which the key save-ans is active.

Detection of the anskey* environment outside the enumext and enumext* environments.

```
2549 \__enumext_before_env:nn { enumext }
       \bool_lazy_and:nnT
         { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
         { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
2554
           \cs_if_free:cF { __scontents_anskey*_env_begin: }
               \msg_error:nnn { enumext } { anskey-env-error } { anskey* }
         }
     }
   \__enumext_before_env:nn { enumext* }
2562
       \bool_lazy_and:nnT
2563
         { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
2564
         { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
2565
2566
           \cs_if_free:cF { __scontents_anskey*_env_begin: }
               \msg_error:nnn { enumext } { anskey-env-error } { anskey* }
         }
2571
```

Detection of the anskey* environment inside the keyans, keyans* and keyanspic environments, if preceded by a not numbered \item or if it is in *math mode* returning the appropriate messages.

```
\__enumext_before_env:nn { anskey* }
       \int_compare:nNnT { \l__enumext_keyans_level_int } = { 1 }
2576
            \msg_error:nnn { enumext } { anskey-env-wrong }{ keyans }
2578
       \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
         {
2580
            \msg_error:nnn { enumext } { anskey-env-wrong } { keyans* }
2581
2582
       \int_compare:nNnT { \l__enumext_keyans_pic_level_int } = { 1 }
2583
2584
            \msg_error:nnn { enumext } { anskey-env-wrong } { keyanspic }
       \bool_if:NF \l__enumext_item_number_bool
2588
            \msg_error:nn { enumext } { anskey-unnumber-item }
2589
         }
2590
       \mode_if_math:T
2591
         {
2592
            \msg_error:nnn { enumext } { anskey-math-mode } { anskey* }
2593
2594
     }
2595
```

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

anskey* _enumext_anskey_env_make:n

__enumext_anskey_env_make:V _enumext_anskey_env_define_keys: __enumext_rescan_anskey_env:n

The function __enumext_anskey_env_make:n creates the environment anskey* (custom version of scontents environment) by setting the initial keys store-env={\store name\}} and print-env=false. To maintain the scope of the environment and that it is only active when the key save-ans is active we will pass this function to the function __enumext_storing_exec: (\s12.25.1) and we will execute it only if the variable \l__enumext_anskey_env_bool is true, with this we prevent it from being executed again when the environment is nested and the key save-ans is active, which returns an error for part of the package scontents.

```
2596 \cs_new_protected:Npn \__enumext_anskey_env_make:n #1
2597 {
2598 \bool_if:NT \l__enumext_anskey_env_bool
2599 {
2600 \newenvsc{anskey*}[store-env=#1,print-env=false]
2601 \__enumext_anskey_env_exec:
2602 }
2603 }
2604 \cs_generate_variant:Nn \__enumext_anskey_env_make:n { V }
```

The function __enumext_anskey_env_define_keys: will add the keys break-col, item-join, item-join, item-star, item-sym* and item-pos* and will leave the keys print-env, store-env and write-out undefined. We will apply this function using the *hook* function __enumext_before_-env:nn.

```
2605 \cs_new_protected:Nn \__enumext_anskey_env_define_keys:
2606
    {
       \keys_define:nn { scontents / scontents }
           break-col .bool_gset:N = \g__enumext_store_columns_break_bool,
          break-col .default:n = true,
          break-col .value_forbidden:n = true,
           item-join .int_gset:N = \g__enumext_store_item_join_int,
2612
           item-join .value_required:n = true,
2613
           item-star .bool_gset:N = \g__enumext_store_item_star_bool,
2614
           item-star .default:n = true,
2615
           item-star .value_forbidden:n = true,
2616
           item-sym* .tl_gset:N = \g__enumext_store_item_symbol_tl,
2617
           item-sym* .value_required:n = true,
2618
           item-pos* .dim_gset:N = \g__enumext_store_item_symbol_sep_dim,
          item-pos* .value_required:n = true,
           print-env .undefine:,
          store-env .undefine:,
          write-out .undefine:,
          unknown .code:n
                                 = { \__enumext_anskey_env_unknown:n {##1} },
2624
2625
    }
2626
```

The $\langle keys \rangle$ are stored in \l_keys_key_str and the value (if any) is passed as an argument to the function _enumext_anskey_env_unknown:n.

```
2627 \cs_new_protected:Npn \__enumext_anskey_env_unknown:n #1
2628 {
       \exp_args:NV \__enumext_anskey_env_unknown:nn \l_keys_key_str {#1}
2629
    }
2630
2631 \cs_new_protected:Npn \__enumext_anskey_env_unknown:nn #1#2
2632
       \tl_if_blank:nTF {#2}
        {
2634
           \msg_error:nnn { enumext } { anskey-env-key-unknown } {#1}
2635
         }
2636
        {
2637
           \msg_error:nnnn { enumext } { anskey-env-key-value-unknown } {#1} {#2}
2638
2639
2640
```

The function __enumext_anskey_env_reset_keys: will leave the keys break-col, item-join, item-join, item-star, item-sym* and item-pos* undefined. We will apply this function using the hook function __enumext_after_env:nn.

```
2641 \cs_new_protected:Nn \__enumext_anskey_env_reset_keys:
    {
2642
       \keys_define:nn { scontents / scontents }
2643
2644
           break-col .undefine:,
2645
           item-join .undefine:,
           item-star .undefine:,
          item-sym* .undefine:,
          item-pos* .undefine:,
           write-out .code:n = {
2650
                                     \bool_set_false:N \l__scontents_storing_bool
2651
                                     \bool_set_true:N \l__scontents_writing_bool
2652
                                     \tl_set:Nn \l__scontents_fname_out_tl {##1}
2653
                                   1.
2654
           write-out .value_required:n = true,
2655
           print-env .meta:nn = { scontents } { print-env = ##1 },
           print-env .default:n = true,
           store-env .meta:nn = { scontents } { store-env = ##1 },
           unknown .code:n = { \__scontents_parse_environment_keys:n {##1} },
         }
2660
2661
```

The function __enumext_rescan_anskey_env:n will be responsible for bringing the $\langle body \rangle$ of the environment saved in the sequence \g__scontents_name_ $\langle store\ name \rangle$ _seq to pass it to our sequence and prop list.

```
262 \cs_new_protected:Npn \__enumext_rescan_anskey_env:n #1
263 {
264    \group_begin:
265    \int_set:Nn \tex_newlinechar:D { `\^^J }
266    \__scontents_rescan_tokens:x
267    {
268        \endgroup % This assumes \catcode`\\=0... Things might go off otherwise.
269        #1
2670    }
2671 }
```

(End of definition for anskey* and others. This function is documented on page 13.)

__enumext_anskey_env_exec:

The function $_$ enumext_anskey_env_exec: will be responsible for processing all the code necessary for the execution of the environment. The first thing will be to add our $\langle keys \rangle$.

Now we will execute our actions after the anskey* environment is closed. We'll fetch the contents of the *environment body* that is now saved in $g_scontents_name_store_name_seq$ and store it in the variable $l_enumext_store_anskey_env_tl$ then we execute the rest of the functions.

```
\hook_if_empty:nF {env/anskey*/after}
           \hook_gremove_code:nn {env/anskey*/after} { * }
2680
2681
       \__enumext_after_env:nn { anskey* }
2682
           \__enumext_anskey_env_save_keys:
           \tl_clear:N \l__enumext_store_anskey_env_tl
           \tl_clear:N \l__enumext_store_anskey_opt_tl
           \bool_if:NT \l__enumext_check_answers_bool
             {
               \tl_gset:Ne \l__enumext_store_anskey_env_tl
                 -{
                   \seq_item:ce { g__scontents_name_ \l__enumext_store_name_tl _seq } { -1 }
                 }
               \regex_match:nVTF
                 { ^s \ z \ ^s \ u\{c\_scontents\_hidden\_space\_str} \ z \ }
                 \l__enumext_store_anskey_env_tl
                   \msg_error:nn { enumext } { anskey-empty-arg }
                 }
                 {
                      _enumext_anskey_env_store:
             _enumext_anskey_env_clean_vars:
           \__enumext_anskey_env_reset_keys:
```

The use of \hook_gremove_code:nn is necessary here, otherwise the {\langle code \rangle} passed to __enumext_after_env:nn{anskey*} will be accumulated for each execution. The last function __enumext_anskey_env_reset_keys: is necessary so as not to hinder any scontents environment running within enumext or enumext*.

(End of definition for __enumext_anskey_env_exec:.)

__enumext_anskey_env_save_keys:
__enumext_anskey_env_store:
__enumext_anskey_env_clean_vars:

The function $_$ enumext_anskey_env_save_keys: processing the $[\langle key = val \rangle]$ passed to the environment and save this in the variable $_$ enumext_store_anskey_opt_tl. If the break-col key is present and the environment is running under enumext (not in enumext*) we will add the key break-col.

If the item-join key is present and the command is running under enumext* we will add to \l_-enumext_store_anskey_opt_tl.

And now we will review the keys item-star, item-sym* and item-pos* and pass them to \l_-enumext_store_anskey_opt_tl.

```
\bool_if:NT \g__enumext_store_item_star_bool
         {
           \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
             {
               ,item-star,
2728
           \tl_if_empty:NF \g__enumext_store_item_symbol_tl
             {
               \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
                    ,item-sym* = \exp_not:V \g__enumext_store_item_symbol_tl,
2734
             7
           \dim_compare:nT
             {
2738
               \g__enumext_store_item_symbol_sep_dim != \c_zero_dim
             }
2740
             {
2741
               \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
                    ,item-pos* = \exp_not:V \g__enumext_store_item_symbol_sep_dim,
                  }
             }
          }
2748
```

The function __enumext_anskey_env_store: will be responsible for storing the content of the environment using the functions __enumext_store_anskey_code:n and __enumext_rescan_anskey_env:n.

```
\cs_new_protected:Nn \__enumext_anskey_env_store:
2749
    {
      \group_begin:
        \tl_if_empty:NTF \l__enumext_store_anskey_opt_tl
          {
           \exp_args:Ne
             \__enumext_store_anskey_code:n
               {
                   _enumext_rescan_anskey_env:n { \l__enumext_store_anskey_env_tl }
2758
         }
2760
            \keys_set_known:nV { enumext / anskey } \l__enumext_store_anskey_opt_tl
2761
           \exp_args:Ne
             \__enumext_store_anskey_code:n
                 }
2767
2768
      \group_end:
```

The function $_$ enumext_anskey_env_clean_vars: will return the global variables used by the $\langle keys \rangle$ to their initial state.

```
2770 \cs_new_protected:Nn \__enumext_anskey_env_clean_vars:
2771 {
2772 \bool_gset_false:N \g__enumext_store_columns_break_bool
2773 \int_gzero:N \g__enumext_store_item_join_int
2774 \bool_gset_false:N \g__enumext_store_item_star_bool
©2024 by Pablo González L
```

```
2775  \tl_gclear:N  \g__enumext_store_item_symbol_tl
2776  \dim_gzero:N  \g__enumext_store_item_symbol_sep_dim
2777  }

(End of definition for \_enumext_anskey_env_save_keys:, \_enumext_anskey_env_store:, and \_enumext_anskey_env_clean vars:)
```

12.31 Executing anskey*, check-ans and write .log

__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, undefine the environment anskey* (§12.30) through the function __enumext_undefine_-anskey_env: and finally we execute the function __enumext_reset_global_vars: returning the used variables to their original state.

```
\cs_new_protected:Nn \__enumext_execute_after_env:
       \int_compare:nNnT { \l__enumext_level_int } = { 0 }
           \tl_if_empty:NF \g__enumext_store_name_tl
                \__enumext_stop_save_ans_msg:
                \__enumext_item_answer_diff:
                \__enumext_log_global_vars:
2786
                \__enumext_log_answer_vars:
2787
                \bool_if:NTF \g__enumext_check_ans_key_bool
2788
                      _enumext_check_ans_show:
                  }
                  { \__enumext_check_ans_log: }
                \verb|\__enumext_undefine_anskey_env:|
             _enumext_reset_global_vars:
2796
2797
```

(End of definition for __enumext_execute_after_env:.)

This function is passed to the function __enumext_after_env:nn for the environments enumext (§12.38) and enumext* (§12.43) and it is executed only when the environments are not nested or at some level of these..

12.32 Common functions for keyans, keyans* and keyanspic

12.32.1 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_current_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.

```
2798 \cs_new_protected:Npn \__enumext_keyans_addto_prop:n #1
2799
       \tl_clear:N \l__enumext_store_current_label_tl
       \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
         {
           \tl_put_right:Ne \l__enumext_store_current_label_tl { \l__enumext_label_vi_tl }
         }
         {
2805
           \tl_put_right:Ne \l__enumext_store_current_label_tl { \l__enumext_label_v_tl }
2806
         }
       \tl_if_novalue:nF { #1 }
           % Set save-sep
           \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
               \tl_put_right:Ne \l__enumext_store_current_label_tl { \l__enumext_store_keyans_item_o
           \tl_put_right:Ne \l__enumext_store_current_label_tl { #1 }
2816
       \__enumext_store_addto_prop:V \l__enumext_store_current_label_tl
2817
©2024 by Pablo González L
                                                                                                 82 / 145
```

```
818
```

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

12.32.2 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 $\langle ref\{\langle store\ name: position \rangle\}$ 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.

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:
2835
       \bool_if:NT \g__enumext_starred_bool
2836
2837
           \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
             { \l__enumext_label_copy_i_tl . \l__enumext_label_copy_vi_tl }
2844
       \int_compare:nNnT { \l__enumext_keyans_level_int } = { 1 }
2845
         {
2846
           \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2847
             { \l__enumext_label_copy_i_tl . \l__enumext_label_copy_v_tl }
2848
       \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 }
2854
       \tl_put_right:Ne \l__enumext_newlabel_arg_one_tl
2855
2856
           \l__enumext_store_name_tl \c_colon_str
2857
           \int_eval:n { \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } }
2858
        \__enumext_keyans_store_ref_aux_ii:
```

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

```
2870 \l__enumext_write_aux_file_tl
2871 }

(End of definition for \__enumext_keyans_store_ref:, \__enumext_keyans_store_ref_aux_i:, and \__enumext_keyans_-
store_ref_aux_ii:.)
```

12.32.3 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_current_label_tl variable to the sequence defined by the save-ans key.

```
2872 \cs_new_protected:Npn \__enumext_keyans_addto_seq:n #1
2873
       \tl_clear:N \l__enumext_store_current_label_tl
2874
       \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
           \tl_put_right:Ne \l__enumext_store_current_label_tl { \item \l__enumext_label_vi_tl }
         }
2878
         {
2879
           \tl_put_right:Ne \l__enumext_store_current_label_tl { \item \l__enumext_label_v_tl }
2881
       \tl_if_novalue:nF { #1 }
         {
           \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
               \tl_put_right:Ne \l__enumext_store_current_label_tl
                 {
                    \l__enumext_store_keyans_item_opt_sep_tl
2888
                 }
           \tl_put_right:Ne \l__enumext_store_current_label_tl { #1 }
       \__enumext_keyans_addto_seq_link:
2893
```

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_current_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 checkans key.

```
2895 \cs_new_protected:Nn \__enumext_keyans_addto_seq_link:
    {
2896
       \bool lazy and:nnT
2897
         { \bool_if_p:N \l__enumext_store_ref_key_bool }
2898
         { \bool_if_p:N \l__enumext_hyperref_bool }
           \tl_put_right:Ne \l__enumext_store_current_label_tl
               \hfill \exp_not:N \hyperlink
                 {
                   \exp_not:V \l__enumext_newlabel_arg_one_tl
                 }
                 { \exp_not:V \l__enumext_mark_ref_sym_tl }
             }
         }
       \__enumext_store_addto_seq:V \l__enumext_store_current_label_tl
2910
       \bool_if:NT \l__enumext_check_answers_bool
2911
           \int_gincr:N \g__enumext_item_anskey_int
         }
```

 $(\textit{End of definition for } \verb|_=enumext_keyans_addto_seq:n | and \verb|_=enumext_keyans_addto_seq_link:|)$

12.32.4 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 \setminus item^* and \langle position \rangle of stored content in \langle prop \ list \rangle for
keyans and keyanspic. Need add 1 to \g__enumext_\( store name \) _prop for show-pos key.
2916 \cs_new_protected:Npn \__enumext_keyans_show_left:n #1
        \tl_if_novalue:nF { #1 }
          {
            \tl_set:Ne \l__enumext_store_current_opt_arg_tl { #1 }
          }
        \bool_if:NT \l__enumext_show_answer_bool
2922
2923
            \__enumext_keyans_show_ans:
2924
2925
        \bool_if:NT \l__enumext_show_position_bool
             \__enumext_keyans_show_pos:
    \cs_new_protected:Nn \__enumext_keyans_show_item_opt:
2931
        \tl_if_empty:NF \l__enumext_store_current_opt_arg_tl
2933
            \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_store_current_opt_arg_tl</mark>        }             \c_space_tl
          }
    \cs_new_protected:Nn \__enumext_keyans_show_ans:
2944
        \bool_if:NT \l__enumext_starred_bool
2945
2946
            \dim_set_eq:NN \l__enumext_labelwidth_i_dim \l__enumext_labelwidth_vii_dim
2947
            \dim_set_eq:NN \l__enumext_labelsep_i_dim \l__enumext_labelsep_vii_dim
2948
2949
        \tl_put_left:Nn \l__enumext_label_v_tl
          {
               _enumext_print_keyans_box:NN
2952
               \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
2953
2954
2955
   \cs_new_protected:Nn \__enumext_keyans_show_pos:
2956
2957
        \bool_if:NT \l__enumext_starred_bool
2958
            \dim_set_eq:NN \l__enumext_labelwidth_i_dim \l__enumext_labelwidth_vii_dim
            \dim_set_eq:NN \l__enumext_labelsep_i_dim \l__enumext_labelsep_vii_dim
        \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
2963
            \tl_set:Ne \l__enumext_mark_answer_sym_tl
2965
2966
                 \group_begin:
                 \exp_not:N \normalfont
                 \exp_not:N \footnotesize [ \int_eval:n
                     \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
                   }
                 \group_end:
          }
2977
            \tl_set:Ne \l__enumext_mark_answer_sym_tl
2978
                 \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
```

(End of definition for __enumext_keyans_show_left:n and others.)

12.33 Redefining \item and \makelabel in enumext

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.

The \item and \item[$\langle custom \rangle$] commands work in the usual way on enumext and we will add \item*, \item*[$\langle symbol \rangle$] and \item*[$\langle symbol \rangle$][$\langle offset \rangle$].

__enumext_default_item:n

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_answers_bool set by the key no-store, set the boolean variable \l__enumext_wrap_label_X_bool to "true" for the key wrap-label and execute __enumext_item_std:w and the key itemindent, otherwise we will check the state of the boolean variable \l__enumext_-wrap_label_opt_X_bool set by the key wrap-label* and execute __enumext_item_std:w with the optional argument and the key itemindent.

```
\cs_new_protected:Npn \__enumext_default_item:n #1
    {
2997
       \tl_if_novalue:nTF {#1}
2998
         {
2999
           \bool_if:NT \l__enumext_check_answers_bool
               \int_gincr:N \g__enumext_item_number_int
               \bool_set_true:N \l__enumext_item_number_bool
           \bool_set_true:c { l__enumext_wrap_label_ \__enumext_level: _bool }
             _enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ \__enumext_level: _tl }
3006
         }
         {
3008
           \bool_set_eq:cc
             { l__enumext_wrap_label_ \__enumext_level: _bool }
3010
             { l__enumext_wrap_label_opt_ \__enumext_level: _bool }
           \__enumext_item_std:w [#1] \tl_use:c { l__enumext_fake_item_indent_ \__enumext_level: _tl
         }
```

(End of definition for __enumext_default_item:n.)

__enumext_starred_item:nn
__enumext_item_star_exec:

The \item*, \item*[$\langle symbol \rangle$] and \item*[$\langle symbol \rangle$] [$\langle offset \rangle$] works like the *numbered* \item, but placing a $\langle symbol \rangle$ to the "*left*" of the $\langle label \rangle$ separated from it by the value 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__enumext_item_symbol_X_tl which is set by the key item-sym* or passed as "first" optional argument in the global variable \g__enumext_item_symbol_aux_tl, followed by setting the variable \l__enumext_item_symbol_sep_X_dim set by the key item-pos* or by the "second" optional argument, then we will see the state of the variable \l__enumext_check_answers_bool set by the key no-store, set the boolean variable \l__enumext_wrap_label_X_bool to "true" for the key wrap-label and execute __enumext_item_std: w and the key itemindent.

©2024 by Pablo González L

__enumext_redefine_item:

__enumext_make_label

```
}
          \tl_if_novalue:nTF {#2}
            {
              \dim set ea:cc
                { l__enumext_item_symbol_sep_ \__enumext_level: _dim }
                { l__enumext_labelsep_ \__enumext_level: _dim }
            {
              \dim_set:cn { l__enumext_item_symbol_sep_ \__enumext_level: _dim } {#2}
  3032
            }
          \bool_if:NT \l__enumext_check_answers_bool
              \int_gincr:N \g__enumext_item_number_int
              \bool_set_true:N \l__enumext_item_number_bool
  3038
          \bool_set_true:c { l__enumext_wrap_label_ \__enumext_level: _bool }
  3039
          \__enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ \__enumext_level: _tl }
  3041
  The function \__enumext_item_star_exec: will be responsible for executing \item* for the enumext
  environment.
  3042 \cs_new_protected:Nn \__enumext_item_star_exec:
          \tl_if_empty:cF { l__enumext_item_symbol_ \__enumext_level: _tl }
  3045
            {
              \mode leave vertical:
  3046
              \skip_horizontal:n { -\dim_use:c { l__enumext_item_symbol_sep_ \__enumext_level: _dim } }
  3047
              \makebox[ Opt ][ r ]{ \g__enumext_item_symbol_aux_tl }
              \skip_horizontal:n { \dim_use:c { l__enumext_item_symbol_sep_ \__enumext_level: _dim } }
  (End of definition for \__enumext_starred_item:nn and \__enumext_item_star_exec:.)
 The function \__enumext_redefine_item: will redefine the \item command in the enumext environ-
  ment adding \item*.
  3052 \cs_new_protected:Nn \__enumext_redefine_item:
  3053
          \RenewDocumentCommand \item { s o o }
  3054
  3055
              \bool_if:nTF {##1}
                   \__enumext_starred_item:nn {##2} {##3}
                { \__enumext_default_item:n {##2} }
  3061
       }
  3062
  The function \__enumext_make_label: redefine \makelabel for the keys align, font, wrap-label,
  wrap-label* and \int m^* for enumext environment.
  3063 \cs_new_protected:Nn \__enumext_make_label:
  3064
          \RenewDocumentCommand \makelabel { m }
  3065
  3066
              \tl_use:c { l__enumext_label_fill_left_ \__enumext_level: _tl }
  3067
              \tl_use:c { l__enumext_label_font_style_ \__enumext_level: _tl }
  3068
              \bool_if:cTF { l__enumext_wrap_label_ \__enumext_level: _bool }
                  \__enumext_item_star_exec:
  3071
                  \use:c { __enumext_wrapper_label_ \__enumext_level: :n } { ##1 }
                }
                { ##1 }
              \tl_use:c { l__enumext_label_fill_right_ \__enumext_level: _tl }
              \tl_gclear:N \g__enumext_item_symbol_aux_tl
  3076
            }
  3077
  (End of definition for \label{local_enum} -\text{enumext\_redefine\_item:} \ and \label{local_enumext_make_label.}
🍼 This functions are passed to \__enumext_list_arg_two_X: used in the definition of the enumext environment
```

©2024 by Pablo González L

(§12.38).

Setting item-sym* and item-pos* keys

In order to have a cleaner implementation of \item* for the enumext and enumext* environments it is best to define a couple of keys that allow us to control and set by default the \(\symbol \) and its \(\choffset \).

```
Define and set item-sym* and item-pos* keys for enumext and enumext*.
item-pos*
            3079 \cs_set_protected:Npn \__enumext_tmp:nn #1 #2
                 {
            3080
                   \keys_define:nn { enumext / #1 }
                       item-sym* .tl_set:c = { l__enumext_item_symbol_#2_tl },
                       item-sym* .value_required:n = true,
            3084
                       item-sym* .initial:n = {$\star$},
                       item-pos* .dim_set:c = { l__enumext_item_symbol_sep_#2_dim },
            2086
                       item-pos* .value_required:n = true,
            3088
            3089
            3090 \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*.)
```

12.35 Handling unknown keys

At this point in the code I already know that I will not add more $\langle keys \rangle$ and since I have already been quite paranoid and restrictive with the definitions of environments and commands, the only thing left to do is do it with the $\langle keys \rangle$ (you have to be consistent in life).

12.35.1 Handling unknown keys for keyans and keyans*

Define and set unknown key for keyans and keyans* environments. unknown

```
\__enumext_keyans_unknown_keys:n
\__enumext_keyans_unknown_keys:nn
```

```
3095 \cs_set_protected:Npn \__enumext_tmp:n #1
3096
     {
       \keys_define:nn { enumext / #1 }
3097
           unknown .code:n = { \__enumext_keyans_unknown_keys:n {##1} }
         }
3102 \clist_map_inline:nn { keyans, keyans* } { \__enumext_tmp:n {#1} }
```

Internal functions for handling unknown key.

```
\cs_new_protected:Npn \__enumext_keyans_unknown_keys:n #1
       \exp_args:NV \__enumext_keyans_unknown_keys:nn \l_keys_key_str {#1}
3107 \cs_new_protected:Npn \__enumext_keyans_unknown_keys:nn #1#2
3108
       \tl_if_blank:nTF {#2}
3109
         {
           \msg_error:nnn { enumext } { keyans-unknown-key } {#1}
         }
         {
           \msg_error:nnnn { enumext } { keyans-unknown-key-value } {#1} {#2}
```

(End of definition for unknown, __enumext_keyans_unknown_keys:n, and __enumext_keyans_unknown_keys:nn.)

12.35.2 Handling unknown keys for enumext*

unknown \ enumext starred unknown keys:n __enumext_starred_unknown_keys:nn

Define and set unknown key for enumext* environment.

```
3117 \keys_define:nn { enumext / enumext* }
3118
       unknown .code:n = { \__enumext_starred_unknown_keys:n {#1} }
3120
```

Internal functions for handling unknown key.

```
3121 \cs_new_protected:Npn \__enumext_starred_unknown_keys:n #1
       \exp_args:NV \__enumext_starred_unknown_keys:nn \l_keys_key_str {#1}
3125 \cs_new_protected:Npn \__enumext_starred_unknown_keys:nn #1#2
©2024 by Pablo González L
```

12.35.3 Handling unknown keys for enumext

3135 \cs_set_protected:Npn __enumext_tmp:n #1

}

3154

3156

unknown

__enumext_standar_unknown_keys:n __enumext_standar_unknown_keys:nn Defines and set the key unknown for enumext environment.

```
{
3136
       \keys_define:nn { enumext / #1 }
            unknown .code:n = { \__enumext_standar_unknown_keys:n {##1} }
3141
   \clist_map_inline:nn { level-1,level-2,level-3,level-4 } { \__enumext_tmp:n {#1} }
Internal functions for handling unknown key.
   \cs_new_protected:Npn \__enumext_standar_unknown_keys:n #1
3144
       \exp_args:NV \__enumext_standar_unknown_keys:nn \l_keys_key_str {#1}
3145
     }
3146
   \cs_new_protected:Npn \__enumext_standar_unknown_keys:nn #1#2
3147
3148
       \tl_if_blank:nTF {#2}
3149
         {
            \msg_error:nnn { enumext } { standar-unknown-key } {#1}
```

 $(End\ of\ definition\ for\ unknown\ ,\ _enumext_standar_unknown_keys:n\ ,\ and\ \setminus_enumext_standar_unknown_keys:nn.)$

\msg_error:nnnn { enumext } { standar-unknown-key-value } {#1} {#2}

12.36 Redefining \item and \makelabel in keyans

The \item and \item[$\langle custom \rangle$] commands work in the usual way in keyans, but the \item* and \item*[$\langle content \rangle$] commands *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.

__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 $\langle label \rangle$.

```
3169 \cs_new_protected:Npn \__enumext_keyans_starred_item:n #1
3170 {
3171  \tl_set_eq:NN \l__enumext_store_current_label_tmp_tl \l__enumext_label_v_tl
3172  \__enumext_keyans_show_left:n { #1 }
3173  \bool_set_true:N \l__enumext_wrap_label_v_bool
3174  \__enumext_item_std:w \tl_use:N \l__enumext_fake_item_indent_v_tl \__enumext_keyans_show_item
32024 by Pablo González L
89/145
```

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 $\langle sequence \rangle$.

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

\item*
__enumext_keyans_redefine_item:
__enumext_keyans_make_label:

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.

```
3181 \cs_new_protected:Nn \__enumext_keyans_redefine_item:
3182
        \RenewDocumentCommand \item { s o }
3183
3184
            \bool_if:nTF {##1}
3185
               {
3186
                 \peek_remove_spaces:n
3187
                   {
                      \__enumext_keyans_starred_item:n {##2}
                   }
              }
              {
                 \__enumext_keyans_default_item:n {##2}
              }
3194
          }
3195
3196
```

The function __enumext_keyans_make_label: redefine \makelabel for the keys align, font, wrap-label, wrap-label* and \item* for keyans environment.

```
\cs_new_protected:Nn \__enumext_keyans_make_label:
     {
3198
       \RenewDocumentCommand \makelabel { m }
           \tl_use:N \l__enumext_label_fill_left_v_tl
3201
           \tl_use:N \l__enumext_label_font_style_v_tl
           \bool_if:NTF \l__enumext_wrap_label_v_bool
3203
             {
                \__enumext_wrapper_label_v:n { ##1 }
             }
             { ##1 }
           \tl_use:N \l__enumext_label_fill_right_v_tl
         }
     }
```

This functions are passed to __enumext_list_arg_two_v: used in the definition of the keyans environment (§12.37.2).

12.37 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.

12.37.1 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 \leftmargin+\itemindent minus \labelwidth+\labelsep. Thus, the handling of the margins by the package will be as shown in the figure 10.

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

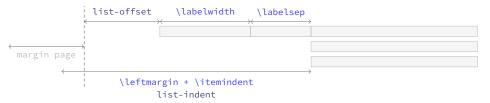


Figure 9: Representation of standard horizontal lengths in list environment.

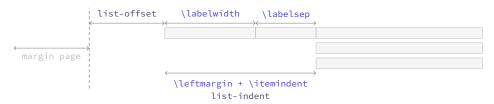


Figure 10: Representation of horizontal lengths concept in list in enumext.

__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 (§12.37.2).

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 }

{

dim_set:Nn #6 { #1 + #2 - #4}

dim_set:Nn #5 { #1 + #2 + #3 - #6 }

}

// dim_set:Nn #5 { #1 + #2 + #3 - #6 }

// dim_set:Nn #5 { #1 + #2 + #3 - #6 }

// dim_set:Nn #5 { \c_zero_dim }

// dim_set:Nn #6 { \c_zero_dim } }

// dim_compare:nNnT { #4 } < { #1 + #2 }

// dim_set:Nn #6 { #1 + #2 - #4} }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:nNnT { #4 } > { #1 + #2 }

// dim_compare:
```



Figure 11: Default horizontal lengths in enumext.

©2024 by Pablo González L 91/145

```
\dim set:Nn #6 { -#1 - #2 + #4}
               \dim_set:Nn #6 { #6*-1}
           \dim_set:Nn #5 { #1 + #2 + #3 - #6 }
3241
3242 \cs_generate_variant:Nn \__enumext_calc_hspace:NNNNNNN { ccccccc }
```

(End of definition for \ enumext calc hspace:NNNNNNN.) 12.37.2 Setting second argument of the lists

```
\__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:
```

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.

```
3243 \cs_set_protected:Npn \__enumext_tmp:n #1
       \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
             _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 }
           \clist_map_inline:nn
             { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
             { \dim_set_eq:cc {####1} { l__enumext_###1_#1_dim } }
           \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
             { \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_redefine_item:
               \__enumext_keyans_make_label:
               \__enumext_keyans_ref:
               \__enumext_keyans_fake_item:
               \bool_if:cT { l__enumext_show_length_#1_bool }
                   \msg_term:nnnn { enumext } { list-lengths-not-nested } { v } { keyans }
                 }
             }
               \__enumext_redefine_item:
               \__enumext_make_label:
               \__enumext_standar_ref:
               \__enumext_fake_item:
               \bool_if:cT { l__enumext_show_length_#1_bool }
                   \msg_term:nnne { enumext } { list-lengths } {#1} { \int_use:N \l__enumext_level_i
                 }
3278
             }
         }
3280
3281
3282 \clist_map_inline:nn { i, ii, iii, iv, v } { \__enumext_tmp:n {#1} }
```

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

__enumext_list_arg_two_vii: __enumext_list_arg_two_viii: 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.

```
3283 \cs_set_protected:Npn \__enumext_tmp:n #1
     {
3284
       \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
3285
3286
           \bool_set_true:c { l__enumext_leftmargin_tmp_#1_bool }
3287
           \dim_zero:c { l__enumext_leftmargin_tmp_#1_dim }
           \__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 }
©2024 by Pablo González L
```

```
\clist_map_inline:nn
             { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
             { \dim_set_eq:cc {####1} { l__enumext_###1_#1_dim } }
           \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
3297
             { \skip_set_eq:cc {####1} { l__enumext_####1_#1_skip } }
3298
           \skip_set_eq:Nc \parsep { l__enumext_itemsep_#1_skip }
3299
           \skip_zero:N \partopsep
3300
           \usecounter { enumX#1 }
3301
           \setcounter { enumX#1 } { \int_eval:n { \int_use:c { l__enumext_start_#1_int } - 1 } }
3302
           \__enumext_starred_ref:
           \str_if_eq:nnTF {#1} { vii }
             {
               \__enumext_fake_item_vii:
               \bool_if:cT { l__enumext_show_length_vii_bool }
                 { \msg_term:nnnn { enumext } { list-lengths-not-nested } { vii } { enumext* } }
3308
             }
3310
               \__enumext_fake_item_viii:
3311
               \bool_if:cT { l__enumext_show_length_#1_bool }
                 { \msg_term:nnnn { enumext } { list-lengths-not-nested } { #1 } { keyans* } }
         }
3317 \clist_map_inline:nn { vii, viii } { \__enumext_tmp:n {#1} }
```

12.38 The environment enumext

enumext We create the enumext environment based on list environment by levels.

(End of definition for __enumext_list_arg_two_vii: and __enumext_list_arg_two_viii:.)

```
3318 \NewDocumentEnvironment{enumext}{ 0{}} }
       \__enumext_safe_exec:
       \__enumext_parse_keys:n {#1}
3321
       \__enumext_before_list:
       \__enumext_start_store_level:
3323
       \__enumext_start_list:nn
         { \tl_use:c { l__enumext_label_ \__enumext_level: _tl } }
         {
3326
            \use:c { __enumext_list_arg_two_ \__enumext_level: : }
3327
            \__enumext_before_keys_exec:
       \__enumext_set_item_width:
       \__enumext_after_args_exec:
     }
3333
       \__enumext_stop_list:
3334
       \__enumext_stop_store_level:
       \__enumext_after_list:
3336
3337
```

(End of definition for enumext. This function is documented on page 4.)

__enumext_set_item_width:

The function __enumext_set_item_width: will set the value of \itemwidth taking into account the value established by the list-offset key for each level of the environment.

```
3338 \cs_new_protected:Nn \__enumext_set_item_width:
     {
       \dim_set:Nn \itemwidth
3340
3341
            \linewidth
3342
         }
3343
       \dim_compare:nT
          {
            \dim_use:c { l__enumext_listoffset_ \__enumext_level: _dim } != \c_zero_dim
3346
          }
3347
          {
            \dim_sub:Nn \itemwidth
3349
                \dim_use:c { l__enumext_listoffset_ \__enumext_level: _dim }
          }
©2024 by Pablo González L
```

(End of definition for __enumext_set_item_width:.)

__enumext_safe_exec:

The __enumext_safe_exec: function first call the function __enumext_internal_mini_page: to create the environment __enumext_mini_env*, then the function __enumext_is_not_nested: which sets \g__enumext_standar_bool to "true" if we are not nested within enumext*, we will increment \l__enumext_level_int to restrict nesting of the environment, set \l__enumext_standar_bool to "true" and finally call the function __enumext_is_on_first_level: which sets \l__enumext_standar_first_bool to "true" only if the environment is not nested and we are at the "first level".

(End of definition for __enumext_safe_exec:.)

__enumext_parse_keys:n

The __enumext_parse_store_keys:n function first we will clear the variable \l__enumext_series_str used by the key series 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 and call the function __enumext_nested_base_line_fix: used by the key base-fix, otherwise we will pass the $\langle keys \rangle$ to the inner levels of the environment then we execute the function __enumext_store_active_keys:n and reprocess the $\langle keys \rangle$ to pass them to the storage $\langle sequence \rangle$ if the key save-key is not active.

```
3366 \cs_new_protected:Npn \__enumext_parse_keys:n #1
     {
3367
       \tl_if_novalue:nF {#1}
3368
3369
           \str_clear:N \l__enumext_series_str
           \int_compare:nNnTF { \l__enumext_level_int } = { 1 }
                \keys_set:nn { enumext / level-1 } {#1}
                \__enumext_parse_series:n {#1}
                \__enumext_nested_base_line_fix:
                \exp_args:Ne \keys_set:nn
3378
                  { enumext / level-\int_use:N \l__enumext_level_int } {#1}
            \__enumext_store_active_keys:n {#1}
         }
3382
3383
```

(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 \(\sequence \) for the command \(\anskey \) and the environment anskey*.

©2024 by Pablo González L 94/145

If enumext are nested in enumext* add __enumext_store_level_open: to preserve the stored structure

```
\bool_lazy_all:nT
3399
         {
3400
           { \bool_if_p:N \l__enumext_store_active_bool }
           { \bool_not_p:n { \l__enumext_keyans_env_bool } }
           { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
         }
         {
3405
           \int_compare:nNnT { \l__enumext_level_int } > { 0 }
3406
             {
3407
                \bool_set_true:c { l__enumext_store_upper_level_ \__enumext_level: _bool }
                \__enumext_store_level_open:
         }
3411
     7
```

Close the stored structure.

 $(\textit{End of definition for } \verb|_= numext_start_store_level: and \verb|_= numext_stop_store_level:.)$

__enumext_before_list:

The function __enumext_before_list: first calls the function __enumext_vspace_above: used by the keys above and above*, then calls the function __enumext_before_args_exec: used by the key before* and finally execute the function __enumext_check_ans_active: for the check answer mechanism.

```
3420 \cs_new_protected:Nn \__enumext_before_list:
3421 {
3422 \__enumext_vspace_above:
3423 \__enumext_before_args_exec:
3424 \__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.

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

__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.

```
3443 \cs_new_protected:Nn \__enumext_multicols_start:
    {
3444
      \int_compare:nNnT
        \dim_compare:nNnT
            { \dim_use:c { l__enumext_columns_sep_ \__enumext_level: _dim } } = { \c_zero_dim }
3450
              \dim_set:cn { l__enumext_columns_sep_ \__enumext_level: _dim }
3451
               {
3452
                 ( \dim_use:c { l__enumext_labelwidth_ \__enumext_level: _dim }
3453
                   + \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 }
               }
            }
          \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
            }
```

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.

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

__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.

```
3473 \cs_new_protected:Nn \__enumext_multicols_stop:
3474
    {
       \int compare:nNnT
3475
         {\int_use:c { l__enumext_columns_ \__enumext_level: _int } } > { 1 }
3476
3477
           \end{multicols}
           \bool_if:cF { l__enumext_minipage_active_ \__enumext_level: _bool }
3479
                \par\addvspace{ \skip_use:c { l__enumext_multicols_below_ \__enumext_level: _skip } }
             }
         }
     }
3484
```

(End of definition for __enumext_multicols_stop:.)

__enumext_after_list:

The function __enumext_after_list: first 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.

```
3485 \cs_new_protected:Nn \__enumext_after_list:
3486 {
3487 \bool_if:cTF { l__enumext_minipage_active_ \__enumext_level: _bool }
3488 {
```

©2024 by Pablo González L

```
\int_compare:nNnT { \g__enumext_minipage_stat_int } = { 1 }

{
    \msg_warning:nn { enumext } { missing-miniright }
    \miniright

}

int_gzero:N \g__enumext_minipage_stat_int

\end{_enumext_mini_env*}

\par\addvspace { \l_enumext_minipage_after_skip }

}

{
_enumext_multicols_stop: }
```

Now we will execute the functions __enumext_after_stop_list: used by the key after, __enumext_-check_ans_key_hook: used by the key check-ans, __enumext_vspace_below: used by the keys below and below*. Finally set \l__enumext_standar_bool to false and call the function __enumext_-resume_save_counter: used by the series, resume and resume* keys.

```
3499 \__enumext_after_stop_list:
3500 \__enumext_check_ans_key_hook:
3501 \__enumext_vspace_below:
3502 \bool_set_false:N \l__enumext_standar_bool
3503 \__enumext_resume_save_counter:
3504 }
```

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.

```
3505 \__enumext_after_env:nn {enumext} { \__enumext_execute_after_env: }

(End of definition for \__enumext_after_list:.)
```

12.39 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".

keyans Now we define the environment keyans also based on lists.

```
3506 \NewDocumentEnvironment{keyans}{ 0{} }
       \__enumext_keyans_safe_exec:
3508
       \__enumext_keyans_parse_keys:n {#1}
       \__enumext_before_list_v:
3510
       \__enumext_start_list:nn
         { \tl_use:N \l__enumext_label_v_tl }
            \__enumext_list_arg_two_v:
3514
            \__enumext_before_keys_exec_v:
3515
         }
3516
       \__enumext_keyans_set_item_width:
       \__enumext_after_args_exec_v:
     }
       \__enumext_check_starred_cmd:n { item }
       \__enumext_stop_list:
3522
       \__enumext_after_list_v:
```

(End of definition for keyans. This function is documented on page 14.)

 $\verb|\| constraints| = constraints| constrain$

The function __enumext_keyans_set_item_width: will set the value of \itemwidth taking into account the value established by the list-offset key.

```
\cs_new_protected:Nn \__enumext_keyans_set_item_width:
     {
3526
        \dim_set:Nn \itemwidth
          {
3528
            \linewidth
          }
        \dim_compare:nT
          {
            \label{local_local_local_local_local} $$ l_e = \c_zero_dim $$
          }
          {
             \dim_sub:Nn \itemwidth
3536
               {
```

```
\l__enumext_listoffset_v_dim
           }
3540
      }
3541
(End of definition for \__enumext_keyans_set_item_width:.)
```

enumext kevans safe exec:

__enumext_before_list_v: \ enumext keyans multicols start:

__enumext_keyans_multicols_stop:

__enumext_after_list_v:

```
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:
                          3543
                                 \bool_if:NF \l__enumext_store_active_bool
                                      \msg_error:nnnn { enumext } { wrong-place }{ keyans }{ save-ans }
                                   }
                                 \int_incr:N \l__enumext_keyans_level_int
                                 \bool_set_true:N \l__enumext_keyans_env_bool
                                 \__enumext_keyans_name_and_start:
                                 % Set false for interfering with enumext nested in keyans (yes, its possible and crayze)
                                 \bool_set_false:N \l__enumext_store_active_bool
                                 \int_compare:nNnT { \l__enumext_keyans_level_int } > { 1 }
                                      \msg_error:nn { enumext } { keyans-nested }
                                   }
                                 \int_compare:nNnT { \l__enumext_level_int } > { 1 }
                          3558
                                      \msg_error:nn { enumext } { keyans-wrong-level }
                          3560
                               }
                          3561
                         (End of definition for \ensuremath{\setminus}_enumext_keyans_safe_exec:.)
\__enumext_keyans_parse_keys:n Parse [\langle key = val \rangle] for keyans environment.
                          3562 \cs_new_protected:Npn \__enumext_keyans_parse_keys:n #1
                               {
                          3563
                                 \keys_set:nn { enumext / keyans } {#1}
                          3564
                          3565
                         (\mathit{End}\ of\ definition\ for\ \verb|\_enumext_keyans_parse_keys:n.)
                         Same implementation as the one used in the enumext environment.
                          3566 \cs_new_protected:Nn \__enumext_before_list_v:
                         3567
                                 \__enumext_vspace_above_v:
                                 \__enumext_before_args_exec_v:
                                 \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
                          3574
                                      \bool_set_true:N \l__enumext_minipage_active_v_bool
                                      \int_gincr:N \g__enumext_minipage_stat_int
                                      \__enumext_keyans_mini_addvspace:
                                      \nointerlineskip\noindent
                                      \begin{__enumext_mini_env*}{ \l__enumext_minipage_left_v_dim }
                                   }
                          3581
                                 \__enumext_keyans_multicols_start:
                          3582
                          3583
                          3584 \cs_new_protected:Nn \__enumext_keyans_multicols_start:
                          3585
                                 \int_compare:nNnT { \l__enumext_columns_v_int } > { 1 }
                          3586
                          3587
                                      \dim_compare:nNnT { \l__enumext_columns_sep_v_dim } = { \c_zero_dim }
```

98 / 145

```
\l__enumext_listoffset_v_dim
             }
           \dim_set_eq:NN \columnsep \l__enumext_columns_sep_v_dim
3598
           \skip_zero:N \multicolsep
           \dim_zero:N \columnseprule % no rule here
           \bool_if:NF \l__enumext_minipage_active_v_bool
                \__enumext_keyans_multi_addvspace:
              }
           \raggedcolumns
           \begin{multicols}{ \l__enumext_columns_v_int }
3607
     }
3608
   \cs_new_protected:Nn \__enumext_keyans_multicols_stop:
3609
3610
       \int_compare:nNnT { \l__enumext_columns_v_int } > { 1 }
3611
         {
3612
            \end{multicols}
3613
           \bool_if:NF \l__enumext_minipage_active_v_bool
                \par\addvspace{ \l__enumext_multicols_below_v_skip }
3618
         }
     }
3619
   \cs_new_protected:Nn \__enumext_after_list_v:
3621
       \bool_if:NTF \l__enumext_minipage_active_v_bool
3622
         {
3623
           \int_compare:nNnT { \g__enumext_minipage_stat_int } = { 1 }
                \msg_warning:nn { enumext } { missing-miniright }
                \miniright
             }
           \int_gzero:N \g__enumext_minipage_stat_int
           \end{__enumext_mini_env*}
3630
            \par\addvspace{ \l__enumext_minipage_after_skip }
3631
         }
3632
         {
3633
              _enumext_keyans_multicols_stop:
         }
3635
       \bool_set_false:N \l__enumext_keyans_env_bool
       \__enumext_after_stop_list_v:
3637
       \__enumext_vspace_below_v:
```

(End of definition for __enumext_before_list_v: and others.)

12.40 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.

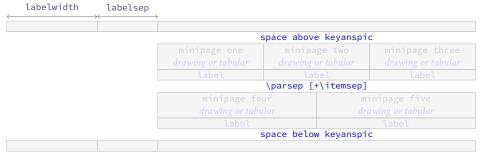


Figure 12: Representation of the keyanspic spacing in enumext.

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?.

12.40.1 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.

```
_{3640} \NewDocumentCommand \anspic { s \ o + m } _{3641} {
```

We check that the command is active in the keyanspic environment only if the save-ans key is present, otherwise we return an error.

```
\bool_if:NF \l__enumext_store_active_bool
           \msg_error:nnnn { enumext } { wrong-place }{ keyanspic }{ save-ans }
2644
3645
       \int_compare:nNnT { \l__enumext_level_int } > { 1 }
3646
         {
3647
           \msg_error:nn { enumext } { keyanspic-wrong-level }
3648
         }
3649
       \int_compare:nNnT { \l__enumext_keyans_level_int } = { 1 }
3651
           \msg_error:nnnn { enumext } { command-wrong-place }{ anspic }{ keyans }
3653
```

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.

```
3654 \seq_put_right:Nn \l__enumext_keyans_pic_body_seq
3655 {
3656 \__enumext_keyans_anspic_code:nnn { #1 } { #2 } { #3 }
3657 }
3658 }
```

(End of definition for \anspic. This function is documented on page 15.)

__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.

```
3659 \cs_new_protected:Nn \__enumext_keyans_anspic_code:nnn
       \stepcounter { enumXvi }
       #3 \\
       \bool_if:nT { #1 }
         {
             _enumext_keyans_addto_prop:n { #2 }
2665
           \__enumext_keyans_store_ref:
2666
           \__enumext_keyans_addto_seq:n { #2 }
3667
           \int_gincr:N \g__enumext_check_starred_cmd_int
3668
           \bool_lazy_or:nnT
             { \bool_if_p:N \l__enumext_show_answer_bool }
3670
             { \bool_if_p:N \l__enumext_show_position_bool }
3671
               \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
2676
3677
       \tl_use:N \l__enumext_label_font_style_v_tl
3678
         _enumext_wrapper_label_v:n { \l__enumext_label_vi_tl } \__enumext_keyans_show_item_opt:
3679
```

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

12.40.2 The environment keyanspic

Now we define the environment keyanspic based on list. The optional argument [\(\number above, number below \)] will determine the number of minipage environments that will be above and below separated by \parsep+\itemsep within it.

©2024 by Pablo González L

```
3687 \__enumext_keyans_pic_arg_two:
```

We apply the "adjusted" vertical spacing above the environment

```
3689 \vspace { \l__enumext_keyans_pic_above_skip }
3600 }
```

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 15.)

__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 __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.

```
3712 \cs_new_protected:Npn \__enumext_keyans_pic_skip_abs:N #1
3713 {
3714 \dim_compare:nNnT { #1 } < { 0pt }
3715 { \skip_set:Nn #1 { -#1 } }
3716 }</pre>
```

(End of definition for $_$ 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.

```
3717 \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 \sep \}
\dim_zero:\n \lambda \listparindent
\dim_zero:\n \lambda \lambda \sep \
\skip_zero:\n \partopsep
\skip_zero:\n \itemsep
```

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
3729
            \box_dp:N \strutbox
            + \l__enumext_topsep_v_skip
            - \parsep
         }
3734
       \__enumext_item_std:w \scan_stop:
       % paranoia
3736
       \RenewDocumentCommand \item {}
3738
            \msg_error:nn { enumext } { keyanspic-item-cmd }
3739
3740
3741
```

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

__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.

```
3742 \cs_new_protected:Nn \__enumext_keyans_pic_do:n
3743 {
3744 \clist_map_function:nN { #1 } \__enumext_keyans_pic_row:n
3745 }
3746 \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.

```
3747 \cs_new_protected:Nn \__enumext_keyans_pic_row:n
3748
       \dim_set:Nn \l__enumext_keyans_pic_width_dim { \linewidth / #1 }
3749
       \int_set:Nn \l__enumext_keyans_pic_above_int { \l__enumext_keyans_pic_below_int }
       \int_set:Nn \l__enumext_keyans_pic_below_int { \l__enumext_keyans_pic_above_int + #1 }
       \int_step_inline:nnn
         { \l__enumext_keyans_pic_above_int + 1 }
         { \l__enumext_keyans_pic_below_int }
             _enumext_minipage:w [ b ]{ \l__enumext_keyans_pic_width_dim }
             \centering
             \seq_item:Nn \l__enumext_keyans_pic_body_seq { ##1 }
             _enumext_endminipage:
         }
3760
       \par
3761
3762
```

(End of definition for $_=$ enumext_keyans_pic_row:n.)

12.41 The horizontal environments

Generating horizontal list environments is NOT as simple as standard LageX 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.

12.42 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 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.

```
3763 \cs_new_protected:Nn \__enumext_footnotetext:nn
3764 {
3765      \footnotetext[#1]{#2}
3766    }
3767 \cs_new_protected:Nn \__enumext_renew_footnote:
3768    {
```

©2024 by Pablo González L

```
\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}
             {
3774
                \stepcounter{footnote}
                \int_gset_eq:Nc \g__enumext_footnote_int { c@footnote }
                \int_gset:Nn \g__enumext_footnote_int { ##1 }
             }
           \footnotemark [ \g__enumext_footnote_int ]
           \seq_gput_right:Nn \g__enumext_footnote_arg_seq { ##2 }
           \seq_gput_right:NV \g__enumext_footnote_int_seq \g__enumext_footnote_int
3783
3784
3785
   \cs_new_protected:Nn \__enumext_print_footnote:
3786
3787
       \seq_if_empty:NF \g__enumext_footnote_int_seq
3788
3789
           \seq_map_pairwise_function:NNN
3790
             \g__enumext_footnote_int_seq
             \g__enumext_footnote_arg_seq
             \__enumext_footnotetext:nn
         }
3794
3795
```

 $(End\ of\ definition\ for\ _enumext_footnotetext:nn\ ,\ _enumext_renew_footnote:\ ,\ and\ _enumext_print_footnote:\)$

12.42.1 Functions for item box width

To achieve the horizontal list environment we will capture the \item command and the content of this in an plain lrbox box using \makebox for the label 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 ($\langle number \rangle$).

enumext starred columns set vii: __enumext_starred_columns_set_viii: We set the default value for the width of the box containing the content of the items for enumext* environment.

```
3796 \cs_new_protected:Nn \__enumext_starred_columns_set_vii:
3797
       \dim_compare:nNnT { \l__enumext_columns_sep_vii_dim } = { \c_zero_dim }
3798
3799
            \dim_set:Nn \l__enumext_columns_sep_vii_dim
                ( \l__enumext_labelwidth_vii_dim + \l__enumext_labelsep_vii_dim )
                / \l__enumext_columns_vii_int
         }
       \int_set:Nn \l__enumext_tmpa_vii_int { \l__enumext_columns_vii_int - 1 }
       \dim_set:Nn \l__enumext_item_width_vii_dim
3807
         {
3808
            ( \linewidth - \l__enumext_columns_sep_vii_dim * \l__enumext_tmpa_vii_int )
3809
            / \l__enumext_columns_vii_int
3810
            - \l__enumext_labelwidth_vii_dim
3811
             \l__enumext_labelsep_vii_dim
3812
When the key rightmargin is active we must adjust the values.
```

```
\dim_compare:nNnT { \l__enumext_rightmargin_vii_dim } > { \c_zero_dim }
         {
            \dim_sub:Nn \l__enumext_item_width_vii_dim
3816
2817
              {
                ( \l__enumext_rightmargin_vii_dim * \l__enumext_tmpa_vii_int )
3818
                / \l__enumext_columns_vii_int
3819
            \dim_add:Nn \l__enumext_columns_sep_vii_dim
                \l__enumext_rightmargin_vii_dim
3823
         }
©2024 by Pablo González L
```

Same implementation for the keyans* environment.

```
3827 \cs_new_protected:Nn \__enumext_starred_columns_set_viii:
3828
       \dim_compare:nNnT { \l__enumext_columns_sep_viii_dim } = { \c_zero_dim }
3829
3830
           \dim_set:Nn \l__enumext_columns_sep_viii_dim
3831
             {
3832
                ( \l__enumext_labelwidth_viii_dim + \l__enumext_labelsep_viii_dim )
3833
                / \l__enumext_columns_viii_int
3834
3835
         }
       \int_set:Nn \l__enumext_tmpa_viii_int { \l__enumext_columns_viii_int - 1 }
       \dim_set:Nn \l__enumext_item_width_viii_dim
           ( \linewidth - \l__enumext_columns_sep_viii_dim * \l__enumext_tmpa_viii_int )
           / \l__enumext_columns_viii_int
3841
           - \l__enumext_labelwidth_viii_dim
3842
            - \l__enumext_labelsep_viii_dim
3843
3844
       \dim_compare:nNnT { \l__enumext_rightmargin_viii_dim } > { \c_zero_dim }
           \dim_sub:Nn \l__enumext_item_width_viii_dim
                ( \l__enumext_rightmargin_viii_dim * \l__enumext_tmpa_vii_int )
                / \l__enumext_columns_viii_int
3851
           \dim_add:Nn \l__enumext_columns_sep_viii_dim
3852
             {
3853
                \l__enumext_rightmargin_viii_dim
3854
3855
         }
3856
     }
3857
```

 $(\textit{End of definition for } \c columns_set_vii: and \c enumext_starred_columns_set_viii:)$

12.42.2 Functions for join item columns

__enumext_starred_joined_item_vii:n
__enumext_starred_joined_item_viii:n

The functions __enumext_starred_joined_item_vii:n and __enumext_starred_joined_item_viii:n will set the *width* of the box in which the content passed to \item($\langle columns \rangle$) will be stored together with the value of \itemwidth for the enumext* environment.

```
3858 \cs_new_protected:Npn \__enumext_starred_joined_item_vii:n #1
3859
     {
       \int_set:Nn \l__enumext_joined_item_vii_int {#1}
3860
       \int_compare:nNnT { \l__enumext_joined_item_vii_int } > { \l__enumext_columns_vii_int }
3861
         {
           \msg_warning:nnee { enumext } { item-joined }
3863
             { \int_use:N \l__enumext_joined_item_vii_int }
             { \int_use:N \l__enumext_columns_vii_int }
           \int_set:Nn \l__enumext_joined_item_vii_int
                   _enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1
3868
3869
3870
       \int_compare:nNnT
3871
         { \l__enumext_joined_item_vii_int }
3872
3873
         { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1 }
           \msg_warning:nnee { enumext } { item-joined-columns }
             { \int_use:N \l__enumext_joined_item_vii_int }
             {
3878
                \int eval:n
3879
                  { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1 }
           \int_set:Nn \l__enumext_joined_item_vii_int
                \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1
       \int_compare:nNnTF { \l__enumext_joined_item_vii_int } > { 1 }
           \int_set_eq:NN \l__enumext_joined_item_aux_vii_int \l__enumext_joined_item_vii_int
©2024 by Pablo González L
                                                                                                 104 / 145
```

```
\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
3905
Same implementation for the keyans* environment.
3907 \cs_new_protected:Npn \__enumext_starred_joined_item_viii:n #1
3908
       \int_set:Nn \l__enumext_joined_item_viii_int {#1}
       \int_compare:nNnT { \l__enumext_joined_item_viii_int } > { \l__enumext_columns_viii_int }
3910
3911
           \msg_warning:nnee { enumext } { item-joined }
3912
             { \int_use:N \l__enumext_joined_item_viii_int }
3913
             { \int_use:N \l__enumext_columns_viii_int }
           \int_set:Nn \l__enumext_joined_item_viii_int
                   _enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1
         }
3919
       \int_compare:nNnT
         { \l__enumext_joined_item_viii_int }
3921
         { \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1 }
         {
           \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 + 1 }
           \int_set:Nn \l__enumext_joined_item_viii_int
3931
             {
3932
               \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1
3933
       \int_compare:nNnTF { \l__enumext_joined_item_viii_int } > { 1 }
           \int_set_eq:NN \l__enumext_joined_item_aux_viii_int \l__enumext_joined_item_viii_int
           \int decr:N \l enumext joined item aux viii int
           \int_add:Nn \l__enumext_item_column_pos_viii_int { \l__enumext_joined_item_aux_viii_int }
           \int_gadd:Nn \g__enumext_item_count_all_viii_int { \l__enumext_joined_item_aux_viii_int }
3941
           \dim_set:Nn \l__enumext_joined_width_viii_dim
               \l__enumext_item_width_viii_dim * \l__enumext_joined_item_viii_int
               + ( \l__enumext_labelwidth_viii_dim + \l__enumext_labelsep_viii_dim
                   + \l__enumext_columns_sep_viii_dim
                 )*\l__enumext_joined_item_aux_viii_int
           \dim_set_eq:NN \itemwidth \l__enumext_joined_width_viii_dim
         }
3951
           \dim_set_eq:NN \l__enumext_joined_width_viii_dim \l__enumext_item_width_viii_dim
3952
           \dim_set_eq:NN \itemwidth \l__enumext_item_width_viii_dim
3953
         }
3954
(\textit{End of definition for $$\_= enumext\_starred\_joined\_item\_vii:n.})
```

©2024 by Pablo González L

12.42.3 Functions for mini-env, mini-right and mini-right* keys

__enumext_start_mini_vii:
__enumext_stop_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.

```
3956 \cs_new_protected:Nn \__enumext_start_mini_vii:
3957
       \dim_compare:nNnT { \l__enumext_minipage_right_vii_dim } > { \c_zero_dim }
           \dim_set:Nn \l__enumext_minipage_left_vii_dim
3960
3961
             {
               \linewidth
               - \l__enumext_minipage_right_vii_dim
               - \l__enumext_minipage_hsep_vii_dim
             }
           \bool_set_true:N \l__enumext_minipage_active_vii_bool
           \dim_gset_eq:NN
             \g__enumext_minipage_right_vii_dim
             \l__enumext_minipage_right_vii_dim
           \__enumext_mini_addvspace_vii:
           \nointerlineskip\noindent
           \begin{__enumext_mini_env*}{ \l__enumext_minipage_left_vii_dim }
3972
3973
```

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_env:nn to execute the __enumext_mini_env* on the "right side".

```
3975 \cs_new_protected:Nn \__enumext_stop_mini_vii:
3976 {
3977     \bool_if:NT \l__enumext_minipage_active_vii_bool
3978     {
3979          \end{__enumext_mini_env*}
3980          \hfill
3981          \bool_gset_true:N \g__enumext_minipage_active_vii_bool
3982     }
```

Finally we execute the $\{\langle code \rangle\}$ passed to the mini-right or mini-right* keys stored in the variable \g__enumext_miniright_code_vii_tl in the __enumext_mini_env* environment on the "right side". For compatibility with the caption package and possibly other $\{\langle code \rangle\}$ passed to this key, we will pass it to a box and then print it.

```
3984 \__enumext_after_env:nn {enumext*}
    {
3985
       \bool_if:NT \g__enumext_minipage_active_vii_bool
3986
           \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
                 \tl_put_left:Nn \g__enumext_miniright_code_vii_tl
                   {
                     \centering
             \vbox_set_top:Nn \l__enumext_miniright_code_vii_box
                 \tl_use:N \g__enumext_miniright_code_vii_tl
             \box_use_drop:N \l__enumext_miniright_code_vii_box
           \end{__enumext_mini_env*}
           \par\addvspace{ \g__enumext_minipage_after_skip }
         }
       \bool_gset_false:N \g__enumext_minipage_active_vii_bool
4005
       \bool_gset_true:N \g__enumext_minipage_center_vii_bool
4006
       \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
```

(End of definition for __enumext_start_mini_vii: and __enumext_stop_mini_vii:.)

```
\__enumext_start_mini_viii:
\__enumext_stop_mini_viii:
```

The implementation of the mini-env, mini-right and mini-right* keys is identical to the one used in the enumext* environment.

```
4011 \cs_new_protected:Nn \__enumext_start_mini_viii:
4012
       \dim_compare:nNnT { \l__enumext_minipage_right_viii_dim } > { \c_zero_dim }
4013
4014
            \dim_set:Nn \l__enumext_minipage_left_viii_dim
4015
                \linewidth
                - \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
4023
              \l__enumext_minipage_right_viii_dim
4024
            \__enumext_mini_addvspace_viii:
4025
            \nointerlineskip\noindent
            \begin{__enumext_mini_env*}{ \l__enumext_minipage_left_viii_dim }
   \cs_new_protected:Nn \__enumext_stop_mini_viii:
4031
        \bool_if:NT \l__enumext_minipage_active_viii_bool
4032
4033
            \end{__enumext_mini_env*}
4034
            \hfill
4035
            \bool_gset_true:N \g__enumext_minipage_active_viii_bool
4036
4037
4038
   \__enumext_after_env:nn {keyans*}
        \bool_if:NT \g__enumext_minipage_active_viii_bool
            \begin{__enumext_mini_env*}{ \g__enumext_minipage_right_viii_dim }
4043
              \par\addvspace { \g__enumext_minipage_right_skip }
              \bool_if:NF \g__enumext_minipage_center_viii_bool
                {
4046
                  \tl_put_left:Nn \g__enumext_miniright_code_viii_tl
                       \centering
                    }
                }
              \vbox_set_top:Nn \l__enumext_miniright_code_viii_box
                {
4053
                  \tl_use:N \g__enumext_miniright_code_viii_tl
4054
4055
              \box_use_drop:N \l__enumext_miniright_code_viii_box
4056
            \end{__enumext_mini_env*}
4057
            \par\addvspace{ \g__enumext_minipage_after_skip }
4058
        \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
     }
(\textit{End of definition for } \verb|\_=enumext_start_mini_viii: and \verb|\_=enumext_stop_mini_viii:|)
```

12.43 The environment enumext*

enumext*

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.

©2024 by Pablo González L

```
\__enumext_start_store_level_vii:
       \__enumext_start_list:nn { }
4072
              enumext list arg two vii:
4073
            \__enumext_before_keys_exec_vii:
         }
4075
       \__enumext_starred_columns_set_vii:
4076
       \item[] \scan_stop:
4077
       \cs_set_eq:NN \__enumext_stop_item_tmp_vii: \noindent
4078
       \cs_set_eq:NN \item \__enumext_start_item_tmp_vii:
     {
4081
4082
       \__enumext_stop_item_tmp_vii:
       \verb|\__enumext_remove_extra_parsep_vii:
4083
       \__enumext_stop_list:
4084
       \__enumext_stop_store_level_vii:
4085
       \__enumext_after_list_vii:
4086
4087
```

(End of definition for enumext*. This function is documented on page 4.)

__enumext_safe_exec_vii:

We will first call the function __enumext_internal_mini_page: to create the environment __enumext_mini_env*, then the function __enumext_is_not_nested: which sets \g__enumext_-starred_bool to true if we are not nested within enumext, we will increment \l__enumext_level_-h_int to restrict nesting of the environment, set \l__enumext_starred_bool to true and finally call the function __enumext_is_on_first_level: which sets \l__enumext_starred_first_bool to true if we are not nested, allowing the "storage system" to be used.

```
4088 \cs_new_protected:Nn \__enumext_safe_exec_vii:
     {
       \__enumext_internal_mini_page:
4090
       \__enumext_is_not_nested:
4091
       \int_incr:N \l__enumext_level_h_int
       \int_compare:nNnT { \l__enumext_level_h_int } > { 1 }
           \msg_error:nn { enumext } { nested }
         }
       \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
         {
4098
           \msg_error:nnn { enumext } { nested-horizontal } { keyans*}
4099
         }
4100
       \bool_set_true:N \l__enumext_starred_bool
4101
       \bool_set_false:N \l__enumext_standar_bool
4102
       \__enumext_is_on_first_level:
```

(End of definition for $\ensuremath{\setminus} _$ enumext $_$ safe $_$ exec $_$ vii:.)

__enumext_parse_keys_vii:n

First we will clear the variable \l__enumext_series_str used by the key series, process the environment $\lceil \langle key = val \rangle \rceil$ and execute the function __enumext_parse_series:n and used by the key series, then we execute the function __enumext_store_active_keys_vii:n and reprocess the $\langle keys \rangle$ to pass them to the storage $\langle sequence \rangle$ if the key save-key is not active and finally we call the function __enumext_nested_base_line_fix: used by the key base-fix.

(End of definition for $_=$ enumext_parse_keys_vii:n.)

__enumext_before_list_vii:

The function __enumext_before_list_vii: first calls the function __enumext_vspace_above_-vii: used by the keys above and above*, then calls the function __enumext_check_ans_active: for the check answer mechanism and finally calls the functions __enumext_before_args_exec: and __enumext_start_mini_vii: used by the keys before*, mini-env, mini-right and mini-right*.

```
4116 \cs_new_protected:Nn \__enumext_before_list_vii:
4117 {
4118 \__enumext_vspace_above_vii:
4119 \__enumext_check_ans_active:
4120 \__enumext_before_args_exec_vii:
4121 \__enumext_start_mini_vii:
4122 }
(End of definition for \__enumext_before_list_vii:.)
```

__enumext_after_list_vii:

The function __enumext_after_list_vii: first calls the function __enumext_stop_mini_vii: used by the keys mini-env, mini-right and mini-right*, then to the functions __enumext_after_stop_list_vii: used by the key after, __enumext_check_ans_key_hook: used by the key check-ans, __enumext_vspace_below_vii: used by the keys below and below*. Finally set \l__enumext_starred_bool to false and call the __enumext_resume_save_counter: function used by the series, resume and resume* keys.

```
4123 \cs_new_protected:Nn \__enumext_after_list_vii:
4124 {
4125 \__enumext_stop_mini_vii:
4126 \__enumext_after_stop_list_vii:
4127 \__enumext_check_ans_key_hook:
4128 \__enumext_vspace_below_vii:
4129 \bool_set_false:N \l__enumext_starred_bool
4130 \__enumext_resume_save_counter:
4131 }
```

(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 and anskey* environment if enumext* are nested in enumext.

(End of definition for __enumext_start_store_level_vii: and __enumext_stop_store_level_vii:.)

12.43.1 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.

__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).

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

__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.

```
4165 \cs_new_protected:Npn \__enumext_joined_item_vii:w (#1)
4166 {
4167 \__enumext_starred_joined_item_vii:n {#1}
4168 \peek_meaning_remove:NTF *
4169 {\__enumext_starred_item_vii:w }
4170 {\__enumext_standar_item_vii:w }
4171 }
```

(End of definition for __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 \ifenoitemarg to true to execute the enumerated version of \item by means of the function __enumext_start_item_vii:w [\l__enumext_label_vii_tl].

```
\cs_new_protected:Npn \__enumext_standar_item_vii:w
       \bool_set_false:N \l__enumext_item_starred_vii_bool
         \peek_meaning:NTF [
           {
             \bool_set_eq:NN
               \l__enumext_wrap_label_vii_bool
               \l__enumext_wrap_label_opt_vii_bool
              \__enumext_start_item_vii:w
           }
           {
              \bool_set_true:N \l__enumext_wrap_label_vii_bool
4182
             \legacy_if_set_true:n { @noitemarg }
4184
              \__enumext_start_item_vii:w [ \l__enumext_label_vii_tl ]
4185
4186
4187
```

 $(\textit{End of definition for } \c\c\c) = \texttt{numext_standar_item_vii:w.})$

__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*[$\langle symbol \rangle$] [$\langle offset \rangle$].

```
4188 \cs_new_protected:Npn \__enumext_starred_item_vii:w
       \bool_set_true:N \l__enumext_item_starred_vii_bool
       \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 }
4194
   \cs_new_protected:Npn \__enumext_starred_item_vii_aux_i:w [#1]
4196
4197
       \tl_gset:Nn \g__enumext_item_symbol_aux_vii_tl {#1}
4198
       \__enumext_starred_item_vii_aux_ii:w
4201 \cs_new_protected:Npn \__enumext_starred_item_vii_aux_ii:w
©2024 by Pablo González L
                                                                                                  110 / 145
```

```
\peek_meaning:NTF [
         { \__enumext_starred_item_vii_aux_iii:w }
4205
         {
           \dim_set_eq:NN
4206
             \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 ]
   \cs_new_protected:Npn \__enumext_starred_item_vii_aux_iii:w [#1]
4214
       \dim_set:Nn \l__enumext_item_symbol_sep_vii_dim {#1}
4215
       \legacy_if_set_true:n { @noitemarg }
4216
       \__enumext_start_item_vii:w [ \l__enumext_label_vii_tl ]
4217
4218
```

(End of definition for __enumext_starred_item_vii:w and others.)

12.43.2 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 __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.

```
\cs_new_protected_nopar:Npn \__enumext_start_item_vii:w [#1]
4220
       \cs_set_eq:NN \__enumext_stop_item_tmp_vii: \__enumext_stop_item_vii:
4221
       \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}
                \bool_if:NT \l__enumext_check_answers_bool
4232
4233
                    \int_gincr:N \g__enumext_item_number_int
                    \bool_set_true:N \l__enumext_item_number_bool
                  }
             }
4237
         }
4238
```

Here we start capturing \item and its contents into a group using the plain form of the \lambda rbox environment. 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.

```
\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} }
               }
               {
                  \makebox[ \l__enumext_labelwidth_vii_dim ][ \l__enumext_align_label_vii_str ]{ #1 }
4267
           \group_end:
           \skip_horizontal:N \l__enumext_labelsep_vii_dim
           \tl_use:N \l__enumext_after_list_args_vii_tl
4270
           \__enumext_minipage:w [ t ]{ \l__enumext_joined_width_vii_dim }
              \skip_set_eq:NN \parindent \l__enumext_listparindent_vii_dim
             \skip_set_eq:NN \parskip \l__enumext_parsep_vii_skip
             \tl_use:N \l__enumext_fake_item_indent_vii_tl
(End of definition for \__enumext_start_item_vii:w.)
```

__enumext_stop_item_vii:

The function __enumext_stop_item_vii: shall terminate with the capture of \item and its \(\chiontents \). Close the environments minipage, \(\text{lrbox} \) and the group. Then we only have to set the width of the box and print it next to \(\text{footnote}, \) and add the horizontal and vertical separation between the boxes.

```
4276 \cs_new_protected_nopar:Nn \__enumext_stop_item_vii:
4277
           \__enumext_endminipage:
4278
         \endlrbox
       \group_end:
       \box_set_wd:Nn \l__enumext_item_text_vii_box
           \l__enumext_joined_width_vii_dim
           + \l enumext labelwidth vii dim
           + \l__enumext_labelsep_vii_dim
4286
       \int_set:Nn \hbadness { 10000 }
4287
       \box_use_drop:N \l__enumext_item_text_vii_box
       \bool_if:NF \l__enumext_footnotes_key_bool
            \__enumext_print_footnote:
         }
       \int_compare:nNnTF { \l__enumext_item_column_pos_vii_int } = { \l__enumext_columns_vii_int }
         {
           \par\noindent
           \int_zero:N \l__enumext_item_column_pos_vii_int
4296
4297
         { \hspace{ \l_enumext_columns_sep_vii_dim } }
4298
```

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

_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.

```
4300 \cs_new_protected:Nn \__enumext_remove_extra_parsep_vii:
       \int_compare:nNnT
         {
            \int_mod:nn { \g__enumext_item_count_all_vii_int } { \l__enumext_columns_vii_int }
         }
4305
4306
         { 0 }
4307
          {
4308
            \vspace{ -\l__enumext_itemsep_vii_skip }
            \int_gzero:N \g__enumext_item_count_all_vii_int
         }
4313
©2024 by Pablo González L
                                                                                                     112/145
```

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.

```
4314 \__enumext_after_env:nn {enumext*} { \__enumext_execute_after_env: }

(End of definition for \__enumext_remove_extra_parsep_vii:.)
```

12.44 The environment keyans*

keyans* 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.

```
\NewDocumentEnvironment{keyans*}{ o }
       \__enumext_safe_exec_viii:
4317
       \__enumext_parse_keys_viii:n {#1}
4318
       \__enumext_before_list_viii:
       \__enumext_start_list:nn { }
           \__enumext_list_arg_two_viii:
           \__enumext_before_keys_exec_viii:
4323
       \__enumext_starred_columns_set_viii:
       \item[] \scan_stop:
       \cs_set_eq:NN \__enumext_stop_item_tmp_viii: \noindent
       \cs_set_eq:NN \item \__enumext_start_item_tmp_viii:
4329
       \__enumext_stop_item_tmp_viii:
       \__enumext_remove_extra_parsep_viii:
4332
       \__enumext_check_starred_cmd:n { item }
4333
       \__enumext_stop_list:
4334
       \__enumext_after_list_viii:
4335
```

(End of definition for keyans*. This function is documented on page 14.)

__enumext_safe_exec_viii: First check the maximum nesting level for the keyans* environment.

```
4337 \cs_new_protected:Nn \__enumext_safe_exec_viii:
       \int_incr:N \l__enumext_keyans_level_h_int
       \int_compare:nNnT { \l__enumext_keyans_level_h_int } > { 1 }
         {
4341
            \msg_error:nn { enumext } { nested }
4342
4343
        \__enumext_keyans_name_and_start:
4344
        \bool_if:NT \l__enumext_starred_bool
4345
4346
            \msg_error:nnn { enumext } { nested-horizontal } { enumext* }
         }
       \bool_set_true:N \l__enumext_starred_bool
       % Set false for interfering with enumext nested in keyans* (yes, its possible and crayze)
       \bool_set_false:N \l__enumext_store_active_bool
        \int_compare:nNnT { \l__enumext_level_int } > { 1 }
          {
            \msg_error:nn { enumext } { keyans-wrong-level }
4354
          }
4355
     }
4356
(End\ of\ definition\ for\ \verb|\__enumext\_safe\_exec\_viii:.)
```

__enumext_parse_keys_viii:n Parse $[\langle key = val \rangle]$ for keyans*.

(End of definition for __enumext_parse_keys_viii:n.)

__enumext_before_list_viii:

The function __enumext_before_list_viii: 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_viii: handle by mini-env.

__enumext_after_list_viii:

The function __enumext_after_list: first call the function __enumext_stop_mini_viii:, 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.

(End of definition for __enumext_after_list_viii:.) 12.44.1 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.

```
4376 \cs_new_protected_nopar:Nn \__enumext_start_item_tmp_viii:
4377 {
4378 \__enumext_stop_item_tmp_viii:
4379 \int_incr:N \l__enumext_item_column_pos_viii_int
4380 \int_gincr:N \g__enumext_item_count_all_viii_int
4381 \__enumext_item_peek_args_viii:
4382 }
```

__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).

(End of definition for __enumext_item_peek_args_viii:.)

(End of definition for $\ensuremath{\setminus}$ enumext_start_item_tmp_viii:.)

 $\verb|__enumext_joined_item_viii:w|$

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.

```
4389 \cs_new_protected:Npn \__enumext_joined_item_viii:w (#1)
4390 {
4391 \__enumext_starred_joined_item_viii:n {#1}
4392 \peek_meaning_remove:NTF *
4393 {\__enumext_starred_item_viii:w }
4394 {\__enumext_standar_item_viii:w }
4395 }
©2024 by Pablo González L
```

(End of definition for __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

```
4396 \cs_new_protected:Npn \__enumext_standar_item_viii:w
4397
       \bool_set_false:N \l__enumext_item_starred_viii_bool
4398
         \peek_meaning:NTF [
           {
             \bool set eq:NN
               \l__enumext_wrap_label_viii_bool
               \l__enumext_wrap_label_opt_viii_bool
             \__enumext_start_item_viii:w
4404
           }
             \bool_set_true:N \l__enumext_wrap_label_viii_bool
             \legacy_if_set_true:n { @noitemarg }
             \__enumext_start_item_viii:w [ \l__enumext_label_viii_tl ]
           }
```

(End of definition for __enumext_standar_item_viii:w.)

__enumext_starred_item_viii:w __enumext_starred_item_viii_aux_i: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_store_current_opt_arg_tl and will save this argument along with the spacing set by the key save-sep in variable \l__enumext_store_current_label_tl if present, then call the function __enumext_starred_item_viii_aux_ii:w.

```
4420 \cs_new_protected:Npn \__enumext_starred_item_viii_aux_i:w [#1]
  4421
                                  \tl_clear:N \l__enumext_store_current_label_tl
  4422
                                  \tl_if_novalue:nF { #1 }
  4423
                                                    \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
                                                                      \tl_put_right:Ne \l__enumext_store_current_label_tl { \l__enumext_store_keyans_item_o
                                                                      \tl_put_right:Ne \l__enumext_store_current_label_tl { #1 }
                                                     \tl_set:Ne \l__enumext_store_current_opt_arg_tl { #1 }
  4430
  4431
                                   \__enumext_starred_item_viii_aux_ii:w
  4432
  4433
                \cs_new_protected:Npn \__enumext_starred_item_viii_aux_ii:w
  4434
  4435
                                  \legacy_if_set_true:n { @noitemarg }
  4436
                                  \__enumext_start_item_viii:w [ \l__enumext_label_viii_tl ]
  4438
(\textit{End of definition for } \_\texttt{enumext\_starred\_item\_viii:w}, \_\texttt{enumext\_starred\_item\_viii\_aux\_i:w}, \textit{and } \_\texttt{enumext\_starred\_item\_viii:w}, \texttt{and } \texttt{and } \texttt{anumext\_starred\_item\_viii:w}, \texttt{anumext\_starred\_item\_vii:w}, \texttt{anumext\_starred\_item\_vii:w}, \texttt{anumext\_starred\_item\_vii:w}, \texttt{a
starred_item_viii_aux_ii:w.)
```

\ enumext starred item exec:

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.

```
4439 \cs_new_protected:Nn \__enumext_starred_item_exec:
    {
       \tl_put_left:Ne \l__enumext_store_current_label_tl { \l__enumext_label_viii_tl }
4441
       \__enumext_store_addto_prop:V \l__enumext_store_current_label_tl
4442
       \__enumext_keyans_store_ref:
4443
       \tl_put_left:Ne \l__enumext_store_current_label_tl { \item }
4444
       \__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
         }
       \verb|\bool_if:NT \l|_enumext\_show\_position\_bool|
4451
4452
           \tl_set:Ne \l__enumext_mark_answer_sym_tl
4453
             {
4454
                \group_begin:
4455
                  \exp_not:N \normalfont
4456
                  \exp_not:N \footnotesize [ \int_eval:n
4457
                      \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
                    }
                   1
               \group_end:
              _enumext_print_keyans_box:NN \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
4464
4465
     }
```

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

12.44.2 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
4484
             {
                \__enumext_renew_footnote:
4486
4487
           \bool_if:NT \l__enumext_item_starred_viii_bool
4488
             {
                \__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
©2024 by Pablo González L
                                                                                                 116 / 145
```

```
}
           \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
4507
             \bool_if:NT \l__enumext_item_starred_viii_bool
4508
                 \tl_use:N \l__enumext_fake_item_indent_viii_tl
                 \__enumext_keyans_show_item_opt:
                 \skip_horizontal:n { -\l__enumext_fake_item_indent_viii_dim - \l__enumext_labelsep_
               }
               {
                 \tl_use:N \l__enumext_fake_item_indent_viii_tl
4515
4517
```

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

__enumext_stop_item_viii:

__enumext_remove_extra_parsep_viii:

The function __enumext_stop_item_viii: shall terminate with the capture of \item and its \(\contents \). 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.

```
4518 \cs_new_protected_nopar:Nn \__enumext_stop_item_viii:
4519
     {
           \__enumext_endminipage:
         \endlrbox
4521
       \group_end:
4522
       \box_set_wd:Nn \l__enumext_item_text_viii_box
4523
4524
           \l__enumext_joined_width_viii_dim
           + \l__enumext_labelwidth_viii_dim
            + \l__enumext_labelsep_viii_dim
         }
       \int_set:Nn \hbadness { 10000 }
       \box_use_drop:N \l__enumext_item_text_viii_box
       \bool_if:NF \l__enumext_footnotes_key_bool
4531
         {
            \__enumext_print_footnote:
4534
       \int_compare:nNnTF
         { \l__enumext_item_column_pos_viii_int } = { \l__enumext_columns_viii_int }
           \par\noindent
           \int_zero:N \l__enumext_item_column_pos_viii_int
4540
         { \hspace{ \l__enumext_columns_sep_viii_dim } }
4541
4542
```

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

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
         {
            \int_mod:nn
              { \g__enumext_item_count_all_viii_int }
              { \l__enumext_columns_viii_int }
4549
         }
         { 0 }
            \vspace{ -\l__enumext_itemsep_viii_skip }
            \int_gzero:N \g__enumext_item_count_all_viii_int
         }
     }
(End of definition for \__enumext_remove_extra_parsep_viii:.)
```

©2024 by Pablo González L

117/145

12.45 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 16.)

__enumext_getkeyans_aux:n

The internal function $\ensuremath{\mbox{\tt _enumext_getkeyans_aux:n}}$ is in charge of *splitting* the $\langle argument \rangle$ using ":". If ":" is omitted it will return an error.

```
4564 \cs_new_protected:Npn \__enumext_getkeyans_aux:n #1
     {
       \str_if_in:nnTF {#1} { : }
4566
         {
4567
            \use:e
4568
              {
4569
                \cs_set:Npn \exp_not:N \__enumext_tmp:w ##1 \c_colon_str ##2 \scan_stop:
                  { {##1} {##2} }
4571
4572
            \exp_after:wN \__enumext_getkeyans:nn \__enumext_tmp:w #1 \scan_stop:
4573
          { \msg_error:nnn { enumext } { missing-colon } {#1} }
4576
```

(End of definition for $_$ 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 $\langle prop\ list \rangle$.

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

12.46 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 $\langle \mathit{keys} \rangle$ for \printkeyans* and will be set by \setenumext[$\langle \mathit{print*} \rangle$] and the variable \l_enumext_print_keyans_vii_tl will have the default keys for the environment enumext* nested within the $\langle \mathit{sequence} \rangle$ and will be set by \setenumext[$\langle \mathit{print*}, * \rangle$], the rest of the variables will be for the environment enumext and will be set by \setenumext[$\langle \mathit{print*}, \mathit{level} \rangle$].

```
4585 \keys_define:nn { enumext / print }
4586
       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 },
                           = \keys_precompile:neN { enumext / level-3 }
       print-3 .code:n
                                { \__enumext_filter_save_key:n {#1} }
©2024 by Pablo González L
                                                                                              118/145
```

```
\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} }
                               \l__enumext_print_keyans_iv_tl,
      print-4 .initial:n = { nosep, label=\Alph*., first=\small, font=\small },
4606
                           = \keys_precompile:neN { enumext / enumext* }
                               { \__enumext_filter_save_key:n {#1} }
                               \l__enumext_print_keyans_vii_tl, % starred nested
      print-* .initial:n = { nosep, label=\arabic*., first=\small, font=\small },
```

 $m{arphi}$ The reason for storing $\langle keys
angle$ in token lists using \keys_precompile:neN is because the keys are set via \setenumext but are later executed by running the command \printkeyans and they are not handled directly by its optional argument, except those related to the first opening level.

\printkeyans Create a user command to print "all stored content" in \(\sequence \) for \anskey, anskey*, \item* and \anspic*. Within a group we will run our "precompiled keys" and then call the internal function __enumext_printkeyans:nnn.

```
4612 \NewDocumentCommand \printkeyans { s O{} m }
4613
       \group_begin:
4614
         \tl_use:N \l__enumext_print_keyans_i_tl
4615
         \tl_use:N \l__enumext_print_keyans_ii_tl
         \tl_use:N \l__enumext_print_keyans_iii_tl
         \tl_use:N \l__enumext_print_keyans_iv_tl
         \tl_use:N \l__enumext_print_keyans_vii_tl
4619
         \__enumext_printkeyans:nnn { #1 } { #2 } { #3 }
       \group_end:
4621
     }
4622
```

(End of definition for \printkeyans. This function is documented on page 16.)

\ enumext printkevans:nnn

The internal function __enumext_printkeyans:nnn will check for the existence of the \(sequence \), if it does not exist it will return an error message, then it will check if not empty.

```
4623 \cs_new_protected:Npn \__enumext_printkeyans:nnn #1 #2 #3
4624
       \seq_if_exist:cTF { g__enumext_#3_seq }
4625
4626
            \seq_if_empty:cF { g__enumext_#3_seq }
4627
4628
                %%\seq_show:c { g__enumext_#3_seq }
```

If the starred argument is present we will check that the environment enumext* is not saved in the (sequence), then execute the variable \l__enumext_print_keyans_starred_tl that contains the default $\langle keys \rangle$ for the environment enumext*, it will open the environment enumext* passing the optional argument to the "first level", set the key base-fix and then will map the \(\sequence \).

```
\bool_if:nTF {#1}
4630
4631
                     \seq_if_in:cnTF { g__enumext_#3_seq } { \end{enumext*} }
4632
4633
                          \label{lem:msg_error:nnnn} $$ \{ enumext \} $$ { print-starred } $$ {\#3} $$ { enumext* } $$
                          \tl_use:N \l__enumext_print_keyans_starred_tl
                          \begin{enumext*}[#2]
                            \keys_set:nn { enumext / level-1 }{ base-fix }
                            \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
                          \end{enumext*}
4641
                   }
```

Otherwise it will open the environment enumext passing the optional argument to the "first level", set the key base-fix and then map the $\langle sequence \rangle$.

```
\begin{enumext}[#2]
                      \keys_set:nn { enumext / enumext* }{ base-fix }
4646
                      \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
4647
                    \end{enumext}
                  }
             }
```

```
4651 }
4652 {
4653 \msg_error:nnn { enumext } { undefined-storage-anskey } {#3}
4654 }
4655 }
```

(End of definition for $_$ enumext_printkeyans:nnn.)

12.47 The command \setenumext

The command \setenumext will be in charge of managing the $\langle keys \rangle$ passed to all environments and to the \printkeyans command. We must take precautions with the enumext* environment and "first level" of the enumext environment so as not to capture $\langle keys \rangle$ that complicate us.

__enumext_filter_first_level:n
__enumext_filter_first_level_key:n
__enumext_filter_first_level_pair:nn

The function __enumext_filter_first_level:n will be in charge of filtering the $\langle keys \rangle$ passed to the environment enumext* and "first level" of the environment enumext.

```
4656 \cs_new:Npn \__enumext_filter_first_level:n #1
4657 {
4658 \use:e
4659 {
4660 \keyval_parse:NNn
4661 \__enumext_filter_first_level_key:n
4662 \__enumext_filter_first_level_pair:nn {#1}
4663 }
4664 }
```

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

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

```
4674 \cs_new:Npn \__enumext_filter_first_level_pair:nn #1#2
4675
        \str_case:nnF {#1}
          {
            { series } {}
4678
            { resume } {}
4679
            { save-ans } {}
4680
          }
4681
          { , { \exp_not:n {#1} } = { \exp_not:n {#2} } }
4682
     }
4683
```

 $(End\ of\ definition\ for__enumext_filter_first_level:n,__enumext_filter_first_level_key:n,\ and__enumext_filter_first_level_pair:nn.)$

Now define a "meta families" of $\langle keys \rangle$ to access from \setenumext.

```
4684 \keys_define:nn { enumext / meta-families }
    {
4685
       enumext-1 .code:n =
                     \keys_set:ne { enumext / level-1 }
                            _enumext_filter_first_level:n {#1}
                   },
       enumext-2 .code:n = { \keys_set:nn { enumext / level-2 } {#1} } ,
4693
       enumext-3 .code:n = { \keys_set:nn { enumext / level-3 } {#1} } ,
4694
       enumext-4 .code:n = { \keys_set:nn { enumext / level-4 } {#1} } ,
4695
                 .code:n = { \keys_set:nn { enumext / keyans
       keyans
                 .code:n =
       enumext*
                     \keys_set:ne { enumext / enumext* }
                         \__enumext_filter_first_level:n {#1}
```

```
},
                                                          keyans*
                                                                              .code:n = { \keys_set:nn { enumext / keyans* } {#1} } ,
                                                                                                                                                                                       = {#1} } } ,
                                                                             .code:n = { \keys_set:nn { enumext / print
                                                                                                                                                                  } { print*
                                                          print*
                                                                                                                                                                  } { print-1 = {#1} } } ,
                                                                             .code:n = { \keys_set:nn { enumext / print
                                                          print-1
                                                                                                                                                                  } { print-2 = {#1} } } ,
                                                                             .code:n = { \keys_set:nn { enumext / print
                                                          print-2
                                                                                                                                                                  } { print-3 = {#1} } } ,
                                                          print-3
                                                                             .code:n = { \keys_set:nn { enumext / print
                                              4708
                                                          print-4
                                                                             .code:n = { \keys_set:nn { enumext / print
                                                                                                                                                                  } { print-4 = {#1} } } ,
                                                                             .code:n = { \keys_set:nn { enumext / print
                                                                                                                                                                 } { print-* = {#1} } } ,
                                                          print-*
                                              4710
                                                          unknown
                                                                             .code:n = { \msg_error:nn { enumext } { unknown-key-family } } ,
                                              4711
                                             We store them in the constant sequence \c__enumext_all_families_seq separated by commas.
                                              \seq_const_from_clist:Nn \c__enumext_all_families_seq
                                             4714
                                                          enumext-1, enumext-2, enumext-3, enumext-4, keyans, enumext*,
                                              4715
                                                          keyans*, print-1, print-2, print-3, print-4, print-*, print*,
                                              4716
                                                      }
                                             Now we define the user command \setenumext.
                     \setenumext
                                              4718 \NewDocumentCommand \setenumext { O{enumext,1} +m }
                                              4719
                                                           \seq_clear:N \l__enumext_setkey_tmpa_seq
                                              4720
                                                           \seq_set_from_clist:Nn \l__enumext_setkey_tmpb_seq {#1}
                                              4721
                                                           \int_set:Nn \l__enumext_setkey_tmpa_int
                                                                  \seq_count:N \l__enumext_setkey_tmpb_seq
                                                           \int_compare:nNnTF { \l__enumext_setkey_tmpa_int } > { 1 }
                                                              {
                                                                  \seq_pop_left:NN \l__enumext_setkey_tmpb_seq \l__enumext_setkey_tmpa_tl
                                              4728
                                                                  \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} }
                                              4736
                                                           \seq_if_empty:NTF \l__enumext_setkey_tmpa_seq
                                              4738
                                                              { \seq_map_inline:Nn \c__enumext_all_families_seq }
                                                              { \seq_map_inline:Nn \l__enumext_setkey_tmpa_seq }
                                              4740
                                                                  \keys_set:nn { enumext / meta-families } { ##1 = {#2} }
                                                              }
                                              4743
                                             (End of definition for \setenumext. This function is documented on page 6.)
 \__enumext_set_parse:n
                                             Internal functions used by the \setenumext command.
\__enumext_set_error:nn
                                              4745 \cs_new_protected:Npn \__enumext_set_parse:n #1
                                                           \tl_set:Ne \l__enumext_setkey_tmpb_tl { \tl_trim_spaces:n {#1} }
                                                          \clist_map_inline:nn { 0, 1, 2, 3, 4, * } % <- max level
                                                              { \tl_remove_all:Nn \l__enumext_setkey_tmpb_tl {##1} }
                                                           \tl_if_empty:NTF \l__enumext_setkey_tmpb_tl
                                                                  \seq_put_right:Ne \l__enumext_setkey_tmpa_seq
                                                                      { \tl_trim_spaces:n {#1} }
                                                               { \__enumext_set_error:nn {#1} { } }
                                              4757 \cs_new_protected:Npn \__enumext_set_error:nn #1 #2
                                                       \{ \mbox{ } \mbox{ }
                                             (End of definition for \__enumext_set_parse:n and \__enumext_set_error:nn.)
```

12.48 The command \setenumextmeta

The command \setenumextmeta will be responsible for adding new "meta-keys" for the enumext and enumext* environments. The implementation code was given by Jonathan P. Spratte (@Skillmon) answer in Add .meta key to existing keys (l3keys).

\setenumextmeta

Internal functions used by the \setenumextmeta command.

__enumext_add_meta_key:nnn
__enumext_def_meta_key:nnn
__enumext_def_meta_key:Vnn

```
4759 \prop_const_from_keyval:Nn \c__enumext_meta_paths_prop
       {enumext,1} = level-1,
4761
       {enumext,2} = level-2,
4762
       {enumext,3} = level-3,
4763
       {enumext,4} = level-4,
       {enumext*} = enumext*
   \NewDocumentCommand \setenumextmeta { s O{enumext,1} m +m }
4768
       \str_if_eq:eeTF { \tl_trim_spaces:n {#3} } { unknown }
4769
         { \msg_error:nn { enumext } { prohibited-unknown } }
           \IfBooleanTF {#1}
             {
4773
                \int_step_inline:nn { 4 }
                 { \__enumext_add_meta_key:nnn { enumext, ##1 } {#3} {#4} }
                \__enumext_add_meta_key:nnn { enumext* } {#3} {#4}
             { \__enumext_add_meta_key:nnn {#2} {#3} {#4} }
         }
4780
4781 \cs_new_protected:Npn \__enumext_add_meta_key:nnn #1
4782
       \tl_set:Nn \l__enumext_meta_path_tl {#1}
4783
       \tl_replace_all:Nnn \l__enumext_meta_path_tl { ~ } {}
4784
       \prop_get:NVNTF
         \c__enumext_meta_paths_prop \l__enumext_meta_path_tl \l__enumext_meta_path_tl
         { \__enumext_def_meta_key:Vnn \l__enumext_meta_path_tl }
         {
           \msg_error:nnn { enumext } { unknown-set } {#1}
           \use none:nn
4790
     }
   \cs_new_protected:Npn \__enumext_def_meta_key:nnn #1#2#3
4793
       \bool_lazy_or:nnTF
         { \keys_if_exist_p:nn { enumext / #1 } {#2} }
4796
         { \keys_if_exist_p:nn { enumext / enumext* } {#2} }
         { \msg_error:nnn { enumext } { already-defined } {#2} }
         {
           \keys_define:nn { enumext / #1 }
4801
                #2 .meta:n = \{ #3 \},
4802
                #2 .value_forbidden:n = true
4803
4804
4805
4807 \cs_generate_variant:Nn \__enumext_def_meta_key:nnn { V }
```

 $(End\ of\ definition\ for\ \end{\ } \cline{A}$ where \cline{A} is documented on page 6.)

12.49 The command \foreachkeyans

The command \foreachkeyans will be responsible for adding new "meta-keys" for the enumext and enumext* environments. The implementation code was given by Jonathan P. Spratte (@Skillmon) answer in Add .meta key to existing keys (l3keys).

We define a set of $\langle keys \rangle$ for command.

```
start
                .int_set:N = \l__enumext_foreach_start_int,
                .value_required:n = true,
       start
                .int_set:N = \l__enumext_foreach_stop_int,
       stop
                .value_required:n = true,
       stop
                .int_set:N = \l__enumext_foreach_step_int,
       step
                .value_required:n = true,
       step
       wrapper .cs_set_protected:Np = \__enumext_foreach_wrapper:n #1,
       wrapper .value_required:n = true,
                .tl_set:N = \l__enumext_foreach_sep_tl,
                .value_required:n = true,
       unknown .code:n = { \__enumext_parse_foreach_keys:n {#1} }
4826 %% Preset kevs
4827 \keys_precompile:nnN { enumext / foreach }
4828
       before={},after={},start=1,step=1,stop=0,wrapper=#1,sep=
4829
4830
     \g__enumext_foreach_default_keys_tl
4831
_{483^2} ‰ Unknow keys
   \cs_new_protected:Npn \__enumext_parse_foreach_keys:nn #1#2
4833
4834
       \tl_if_blank:nTF {#2}
4835
         { \msg_error:nnn { enumext } { for-key-unknown } {#1} }
         { \mbox{ msg\_error:nnnn } { \mbox{ enumext } { \mbox{ for-key-value-unknown } {#1} {#2} } }
4837
4838
4839 \cs_new_protected:Npn \__enumext_parse_foreach_keys:n #1
4840
       \exp_args:NV \__enumext_parse_foreach_keys:nn \l_keys_key_str {#1}
4841
     }
4842
4843 \NewDocumentCommand \foreachkayans { +O{} m }
       \__enumext_foreach_kayans:nn {#1} {#2}
     }
4847 \cs_new_protected:Npn \__enumext_foreach_kayans:nn #1 #2
4848
       \tl_use:N \g__enumext_foreach_default_keys_tl
4849
       \keys_set:nn { enumext / foreach } {#1}
4850
       \tl_set:Nn \l__enumext_foreach_name_prop_tl {#2}
4851
       \seq_clear:N \l__enumext_foreach_print_seq
4852
       \int_compare:nNnT { \l__enumext_foreach_stop_int } = { 0 }
4853
4854
           \int_set:Nn \l__enumext_foreach_stop_int
4855
              { \prop_count:c { g_enumext_#2_prop } }
       \int_step_function:nnnN
4858
         { \l__enumext_foreach_start_int }
         { \l__enumext_foreach_step_int }
4860
         { \l__enumext_foreach_stop_int }
4861
         \__enumext_foreach_add_body:n
4862
         \seq_use:NV \l__enumext_foreach_print_seq \l__enumext_foreach_sep_tl
4863
     }
   \cs_new_protected:Npn \__enumext_foreach_add_body:n #1
       \seq_put_right:Ne \l__enumext_foreach_print_seq
4868
           \exp_not:V \l__enumext_foreach_before_tl
           \__enumext_foreach_wrapper:n
4870
4871
                \prop_item:cn { g__enumext_ \l__enumext_foreach_name_prop_tl _prop }{#1}
4872
4873
            \exp_not:V \l__enumext_foreach_after_tl
4874
4875
     }
```

12.50 Messages

Message used by package-load for multicol and hyperref packages.

```
The ~ '#1' ~ package ~ will ~ be ~ loaded ~ as ~ a ~ dependency.
4885 \msg_new:nnn { enumext } { package-load-foot }
       The ~ '#1' ~ package ~ is ~ loaded ~ with ~ the ~ option ~ '#2'.
4887
Message used in the creation of counters by enumext package.
4889 \msg_new:nnn { enumext } { counters }
       The ~ counter ~ '#1' ~ is ~ already ~ defined ~ by ~ some ~ \\
       package ~ or ~ macro, ~ it ~ cannot ~ be ~ continued.
4893
Message used by align and mark-pos keys.
4894 \msg_new:nnn { enumext } { unknown-choice }
       The ~ value ~ '#3' ~ for ~ '#1' ~ key ~ is ~ invalid ~ use ~ ('#2').
Message used by reserved anskey* environment by enumext package.
4898 \msg_new:nnnn { enumext } { anskey-env-error }
       The ~ '#1' ~ environment ~is ~ reserved ~ by ~\\
       'enumext' ~ package, ~ It~ is~ already~ defined.
       The ~ anskey* ~ environment ~ is ~ defined ~ internally ~
       for ~ the ~ 'save-ans' ~ key.\\
Message used in the creation of \langle prop \ list \rangle by enumext package.
4907 \msg_new:nnn { enumext } { store-prop }
        * ~ Package ~ enumext: ~ Creating ~
        \c_backslash_str g__enumext_#1_prop ~ \msg_line_context:.
4911
4912 \msg_new:nnn { enumext } { store-seq }
4913
       * ~ Package ~ enumext: ~ Creating ~
4914
       \c_backslash_str g__enumext_#1_seq ~ \msg_line_context:.
4915
4916
4917 \msg_new:nnn { enumext } { store-int }
       * ~ Package ~ enumext: ~ Creating ~
       \c_backslash_str g__enumext_resume_#1_int ~ \msg_line_context:.
4921
4922 \msg_new:nnn { enumext } { prop-seq-int-hook }
4923
       * ~ Package ~ enumext: ~ Elements ~ in ~
4924
      \c_backslash_str g__enumext_#1_prop ~ = ~ #2.\\
       * ~ 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.
4931 \msg_new:nnn { enumext } { item-answer-hook }
       * ~ Package ~ enumext: ~ Value ~ off ~
4933
       \c_backslash_str g__enumext_item_number_int ~ = ~ #1.\\
4934
       * ~ Package ~ enumext: ~ Value ~ off ~
       \c_backslash_str g__enumext_item_anskey_int ~ = ~ #2.\\
4936
       * ~ Package ~ enumext: ~ Difference ~ item_number_int ~ - ~ item_anskey_int ~ = ~ #3.
4937
Message used by [\langle key = val \rangle] system and \setenumext command.
4939 \msg_new:nnn { enumext } { invalid-key }
       The ~ key ~ '#1' ~ is ~ not ~ know ~ the ~ level ~ #2.
4942
4943 \msg_new:nnn { enumext } { unknown-key-family }
```

```
Unknown~key~family~`\l_keys_key_str'~for~enumext.
     }
Messages used in length calculation.
4947 \msg_new:nnn { enumext } { width-negative }
       Ignoring ~ negative ~ value ~ '#1=#2' ~ \msg_line_context:.\\
       The \sim key \sim '#1'\sim accepts \sim values \sim >= \sim Opt.
4951
4952 \msg_new:nnn { enumext } { width-zero }
4953
      Invalid ~ '#1=#2' ~ \msg_line_context:.\\
4954
       The ~ key ~ '#1'~ accepts ~ values ~ > ~ Opt.
4955
4956
Messages used by show-length key in enumext.
4957 \msg_new:nnn { enumext } { list-lengths }
4958
       **** ~ Lengths ~ used ~ by ~ 'enumext' ~ level ~ '#2' ~ \msg_line_context:~\c_space_tl ****\\
       \__enumext_show_length:nnn { dim } { labelsep
                                                          } {#1}
       \__enumext_show_length:nnn { dim } { labelwidth
       \__enumext_show_length:nnn { dim } { itemindent
                                                           } {#1}
       \__enumext_show_length:nnn { dim } { leftmargin } {#1}
       \__enumext_show_length:nnn { dim } { rightmargin } {#1}
       \__enumext_show_length:nnn { dim } { listparindent } {#1}
       \__enumext_show_length:nnn { skip } { topsep
                                                      } {#1}
4966
       \__enumext_show_length:nnn { skip } { parsep
4967
                                                       } {#1}
       \__enumext_show_length:nnn { skip } { partopsep } {#1}
4968
       \__enumext_show_length:nnn { skip } { itemsep } {#1}
4969
4970
4971
Messages used by show-length key in enumext*, keyans* and keyans.
4972 \msg_new:nnn { enumext } { list-lengths-not-nested }
4973
       **** ~ Lengths ~ used ~ by ~ '#2' ~ environment ~ \msg_line_context:~\c_space_tl ****\\
4974
       \__enumext_show_length:nnn { dim } { labelsep
                                                           } {#1}
4975
       \__enumext_show_length:nnn { dim } { labelwidth
4976
       \__enumext_show_length:nnn { dim } { itemindent
4977
       \__enumext_show_length:nnn { dim } { leftmargin
                                                           } {#1}
       \__enumext_show_length:nnn { dim } { listparindent } {#1}
       \__enumext_show_length:nnn { skip } { topsep
       \__enumext_show_length:nnn { skip } { parsep
                                                      } {#1}
4982
       \__enumext_show_length:nnn { skip } { partopsep } {#1}
4983
       \__enumext_show_length:nnn { skip } { itemsep } {#1}
4984
4985
Messages used by ref key.
4987 \msg_new:nnn { enumext } { key-ref-empty }
       Key ~ 'ref' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
Messages used by save-ans key.
4991 \msg_new:nnn { enumext } { save-ans-empty }
       Key ~ 'save-ans' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
4995 \msg_new:nnn { enumext } { save-ans-log }
        ~ Package ~ enumext: ~ Start ~ #1\c_space_tl with ~ save-ans=#2 ~ \msg_line_context:.
4999 \msg_new:nnn { enumext } { save-ans-log-hook }
         ~ Package ~ enumext: ~ Stop ~ #1\c_space_tl with ~ save-ans=#2 ~ \msg_line_context:.
5001
5002
5003 \msg_new:nnn { enumext } { save-ans-hook }
5004
       Stop ~ storing ~ for ~ 'save-ans=#1' ~ \msg_line_context:.
5005
```

Messages used by the internal system to check answer used by check-ans key.

```
5007 \msg_new:nnn { enumext } { need-save-ans }
       Key \sim '#1'\sim works \sim only \sim with \sim the \sim 'save-ans' \sim key \sim in \sim '#2'\sim \msg_line_context:.
5009
5010
5011 \msg_new:nnn { enumext } { items-same-answer }
5012
       ***********
5013
       * ~ Package ~ enumext: ~ Checking ~ answers ~ in ~ '#1' ~
5014
       for ~ \c_left_brace_str #2 \c_right_brace_str\\
5015
       * ~ started ~ #3 ~ and ~ close ~ \msg_line_context: : ~
       'OK', ~ all ~ items ~ with ~ answer.\\
5020 \msg_new:nnn { enumext } { item-greater-answer }
5021
       Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\\
5022
       started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\\
5023
       Items ~ > ~ Answers.
5026 \msg_new:nnn { enumext } { item-less-answer }
       Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\\
5028
       started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\\
       Items ~ < ~ Answers.
5031
Messages used by the internal system to check for "starred" \item* and \anspic* commands.
5932 \msg_new:nnn { enumext } { missing-starred }
5033
       Missing ~ '\c_backslash_str #1*' ~ #2.
5034
5035
5036 \msg_new:nnn { enumext } { many-starred }
5037
       Many ~ '\c_backslash_str #1*' ~ #2.
5039
Messages used by \printkeyans* command.
5040 \msg_new:nnn { enumext } { print-starred }
5041
       \c_{backslash\_str} printkeyans*:~ The ~ sequence ~ '#1' ~ already ~ contains ~
       #2 ~ environment ~ \msg_line_context:.
Message for the nesting depth of the environment enumext.
5045 \msg_new:nnn { enumext } { list-too-deep }
       Too ~ deep ~ nesting ~ for ~ 'enumext' ~ \msg_line_context:.~ \\
       The ~ maximum ~ level ~ of ~ nesting ~ is ~ 4.
Messages used by \anskey, anskey* and \anspic commands.
5050 \msg_new:nnn { enumext } { anskey-unnumber-item }
5051
       Can't ~ store ~ with ~ a ~ unnumbered ~ \c_backslash_str item ~ \msg_line_context:.
5052
5053
5054 \msg_new:nnn { enumext } { anskey-already-stored }
5055
       Content ~ already ~ stored ~ for ~ this ~ \c_backslash_str item ~ \msg_line_context:.
5056
     }
5057
5058 \msg_new:nnn { enumext } { anskey-empty-arg }
       Can't ~ store ~ empty ~ content ~ \msg_line_context:.
5060
5061
5062 \msg_new:nnn { enumext } { anskey-wrong-place }
5063
       Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:.~ \\
5064
       '\c_backslash_str #1' ~ works ~ in ~ the ~ environment ~ '#2'.
5065
5067 \msg_new:nnn { enumext } { anskey-nested }
       The ~ command ~ \c_backslash_str anskey~ can't ~ be ~ nested ~ \msg_line_context:.
©2024 by Pablo González L
```

```
5071 \msg_new:nnn { enumext } { anskey-math-mode }
       #1 ~ can't ~ work ~ in ~ math ~ mode ~ \msg_line_context:.
5074
5075 \msg new:nnn { enumext } { anskev-env-wrong }
       The ~ environment ~ anskey* ~ cannot ~ use ~ in ~ '#1' ~ \msg_line_context:.
5078
   \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'.
5083
5084 \msg_new:nnn { enumext } { command-wrong-place }
5085
       Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:.~ \\
5086
        \c_backslash_str #1' ~ works ~ outside ~ the ~ environment ~ '#2'.
5087
5088
   \msg_new:nnnn { enumext } { anskey-env-key-unknown }
       The \sim key \sim '#1' \sim is \sim unknown \sim by \sim environment\sim
       'anskey*' ~ and ~ is ~ being ~ ignored.
       The ~ environment ~ 'anskey*' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5095
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5097
5098 \msg_new:nnnn { enumext } { anskey-env-key-value-unknown }
5099
       The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~
5100
       'anskey*' ~ and ~ is ~ being ~ ignored.
       The ~ environment ~ 'anskey*' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5106
5107 \msg_new:nnnn { enumext } { anskey-cmd-key-unknown }
     { The ~ key ~'#1'~ is ~ unknown ~ by ~ '\c_backslash_str anskey' ~ and ~ is ~ being ~ ignored.}
5108
5109
       The ~ command ~'\c_backslash_str anskey' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
   \msg_new:nnnn { enumext } { anskey-cmd-key-value-unknown }
     { The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ '\c_backslash_str anskey' ~ and ~ is ~ being ~ igno
       The ~ command ~ '\c_backslash_str anskey' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5116
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
Messages used by keyans, keyans* and keyanspic environment.
silp \msg_new:nnn { enumext } { keyans-nested }
       The ~ environment ~ 'keyans' ~ can't ~ be ~ nested ~ \msg_line_context:.
5123 \msg_new:nnn { enumext } { keyans-wrong-level }
5124
       Wrong ~ level ~ position ~ for ~ 'keyans' ~ \msg_line_context:.~ \\
       The ~ environment ~ 'keyans' ~ can ~ only ~ be ~ in ~ the ~ first ~ level.
5128 \msg_new:nnn { enumext } { wrong-place }
       Wrong ~ place ~ for ~ '#1' ~ environment ~\msg_line_context:.~ \\
5130
       '#1' \sim is \sim only \sim found \sim with \sim '#2' \sim in \sim 'enumext.
5133 \msg_new:nnn { enumext } { keyanspic-nested }
5134
       The ~ environment ~ 'keyanspic' ~ can't ~ be ~ nested~ \msg_line_context:.~.
5136
5137 \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.
```

```
5142 \msg_new:nnn { enumext } { keyanspic-item-cmd }
       Can't ~ use ~ \c_backslash_str item ~ in ~ keyanspic ~ \msg_line_context:.
5144
   \msg_new:nnnn { enumext } { keyans-unknown-key }
5147
       The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment~
        '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5149
       The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
       ~ have ~ a ~ key ~ called ~'#1'.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
   \msg_new:nnnn { enumext } { keyans-unknown-key-value }
5156
       The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~
5158
        '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
       The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
       ~ have ~ a ~ key ~ called ~'#1'.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5164
5165
Message used by unknown \langle keys \rangle in enumext*. environment.
5166 \msg_new:nnnn { enumext } { starred-unknown-key }
       The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment~
5168
        '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
       The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
       ~ have ~ a ~ key ~ called ~'#1'.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
   \msg_new:nnnn { enumext } { starred-unknown-key-value }
       The \sim key \sim '#1=#2' \sim is \sim unknown \sim by \sim environment \sim
5178
        '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5180
5181
       The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5182
       ~ have ~ a ~ key ~ called ~'#1'.\\
5183
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5184
5185
Message used by unknown \langle keys \rangle in enumext environment.
s186 \msg_new:nnnn { enumext } { standar-unknown-key }
5187
       The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment ~ '\l__enumext_envir_name_tl' \c_space_tl
5188
       ~ on ~ level ~ \int_use:N \l__enumext_level_int \c_space_tl and ~ is ~ being ~ ignored.
5189
5190
       The ~ environment ~ '\l_enumext_envir_name_tl' ~ does ~ not
       ~ have ~ a ~ key ~ called ~'#1' ~ on ~ level ~ \int_use:N \l__enumext_level_int.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
   \msg_new:nnnn { enumext } { standar-unknown-key-value }
5196
       The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~ '\l_enumext_envir_name_tl' \c_space_
5198
       ~ on ~ level ~ \int_use:N \l__enumext_level_int \c_space_tl and ~ is ~ being ~ ignored.
5199
       The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
       ~ have ~ a ~ key ~ called ~'#1' ~ on ~ level ~ \int_use:N \l__enumext_level_int.\\
       Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
Message used by unknown \langle keys \rangle in \foreachkeyans.
5206 \msg_new:nnnn { enumext } { for-key-unknown }
```

```
{ The~key~'#1'~is~unknown~by~'\c_backslash_str foreachkeyans'~and~is~being~ignored.}
       The~command~'\c_backslash_str foreachkeyans'~does~not~have~a~key~called~'#1'.\\
       Check~that~you~have~spelled~the~key~name~correctly.
   \msg_new:nnnn { enumext } { for-key-value-unknown }
     { The~key~'#1=#2'~is~unknown~by~'\c_backslash_str foreachkeyans'~and~is~being~ignored. }
       The~command~'\c_backslash_str foreachkeyans'~does~not~have~a~key~called~'#1'.\\
       Check~that~you~have~spelled~the~key~name~correctly.
Messages used by \getkeyans command.
5218 \msg_new:nnn { enumext } { undefined-storage-anskey }
       Storage ~ named ~ '#1' ~ is ~ not ~ defined ~ \msg_line_context:.
Messages used by \miniright command.
   \msg_new:nnn { enumext } { missing-miniright }
       Missing ~ '\c_backslash_str miniright' ~ in ~ \msg_line_context:.\\
       The ~ key ~ 'mini-env' ~ need ~ '\c_backslash_str miniright'.
   \msg_new:nnn { enumext } { wrong-miniright-place }
5228
       Wrong ~ place ~ for ~ '\c_backslash_str miniright' ~ \msg_line_context:.~ \\
       Works ~ in ~ 'enumext' ~ and ~ 'keyans' ~ with ~ key ~ 'mini-env'.
5230
5231
   \msg_new:nnn { enumext } { wrong-miniright-use }
5232
       Wrong ~ use ~ for ~ '\c_backslash_str miniright' ~ \msg_line_context:.~ \\
5234
       '\c_backslash_str miniright' ~ need ~ a ~ key ~ 'mini-env'.
5235
   \msg_new:nnn { enumext } { wrong-miniright-starred }
       Can't ~ use ~ \c_backslash_str miniright ~ in ~ starred ~ environments ~ \msg_line_context:.
5239
5240
5241 \msg_new:nnn { enumext } { many-miniright-used }
5242
       Can't ~ use ~ \c_backslash_str miniright ~ more ~ than ~ once ~ \msg_line_context:.
5243
Messages used by \setenumextmeta command.
   \msg_new:nnn { enumext } { unknown-set }
       Argument ~ [#1] ~ is ~ unknown ~ by ~ \c_backslash_str setenumextmeta ~ \msg_line_context:.
5248
   \msg_new:nnn { enumext } { already-defined }
       The ~ key ~ '#1' ~ is ~ already ~ defined ~ \msg_line_context:.
   \msg_new:nnn { enumext } { prohibited-unknown }
       The ~ name ~ 'unknown' ~ can't ~ be ~ chosen~ for ~ a ~ meta ~ key ~ \msg_line_context:.
5256
Messages used by enumext* and keyans* environments.
5257 \msg_new:nnn { enumext } { nested }
5258
       The ~ environment ~ \l_enumext_envir_name_tl \c_space_tl can't ~ be ~ nested ~ \msg_line_con
5261 \msg_new:nnn { enumext } { nested-horizontal }
       The ~ environment ~ \l__enumext_envir_name_tl \c_space_tl can't ~ be ~ nested ~ in ~ '#1' ~
5263
5265 \msg_new:nnn { enumext } { item-joined }
       Items ~ joined ~ (#1) ~ > ~ #2 ~ columns ~\msg_line_context:.
5269 \msg new:nnn { enumext } { item-joined-columns }
       Not ~ space ~ to ~ join ~ items ~ (#1) ~ > ~ #2 ~\msg_line_context:.
5271
©2024 by Pablo González L
```

5272 }

12.51 Finish package

Finish package implementation.

```
_{5273} \file_input_stop: _{5274} \langle/package\rangle
```

13 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_lazy_and:nnTF 235, 245, 830, 841, 1421, 1799,
* 218	1808, 1972, 1978, 2367, 2374, 2408, 2551, 2563, 2709,
\+ 210	2715, 2897
\ 210	\bool_lazy_or:nnTF 1861, 1868, 2935, 3669, 4795
\\ 226, 2668, 3662, 4891, 4900, 4905, 4925, 4927, 4934, 4936,	\bool_new:N 34, 35, 36, 37, 38, 39, 40, 41, 64, 73, 94, 99,
4949, 4954, 4959, 4974, 5013, 5015, 5017, 5022, 5023,	100, 105, 106, 109, 134, 135, 143, 144, 149, 151, 152,
5028, 5029, 5047, 5064, 5081, 5086, 5095, 5104, 5110,	166, 178, 180
5116, 5125, 5130, 5139, 5153, 5163, 5173, 5183, 5193,	\bool_not_p:n 236, 246, 2305, 2369, 2375, 2711, 2716,
5203, 5209, 5215, 5224, 5229, 5234	3389, 3402
	\bool_set_eq:NN 3009, 3165, 4177, 4401
A	\bool_set_false:N 408, 852, 1906, 1907, 1939, 1944,
above	1948, 1952, 1965, 2651, 3363, 3502, 3552, 3636, 3701,
above* $\dots \underline{1483}$	3719, 4102, 4129, 4174, 4351, 4398
\addvspace 1120, 1148, 1264, 1343, 1406, 1412, 1448, 1470,	\bool_set_true:N . 263, 264, 278, 279, 390, 394, 501,
3481, 3496, 3616, 3631, 3989, 4003, 4044, 4058	867, 1489, 1494, 1756, 1878, 1879, 2151, 2159, 2652,
after 959	3003, 3005, 3037, 3039, 3161, 3173, 3287, 3362, 3395,
align 508	3408, 3434, 3549, 3576, 3966, 4021, 4101, 4183, 4190,
\Alph	4191, 4235, 4349, 4407, 4414, 4415
\Alph	box commands:
\alph 36, 41	\box_dp:N 1160, 1164, 1168, 1179, 1183, 1194, 1203,
\alph 461, 573, 4598	1209, 1219, 1232, 1238, 1244, 1275, 1276, 1277, 1280,
\anskey	1290, 1294, 1303, 1310, 1315, 1323, 1352, 1353, 1356,
anskey*	1363, 1376, 1384, 1390, 1398, 3731 \box_new:N 70, 173, 179
\anspic	\box_set_wd:\n
\anspic* 67	\box_use_drop:\N \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \
\arabic 30, 36	\box_wd:N
\arabic 459, 572, 619, 4590, 4594, 4610	
n.	С
B	\c 218, 219, 725, 727, 739, 741
base-fix 818 \baselineskip 50	\catcode 2668
\baselineskip	\cB
before	\centering 1451, 1473, 3757, 3994, 4049
before*	check-ans
below	Document class:
below* 1483	article 42
bool commands:	clist commands:
\bool_gset_false:N 331, 332, 333, 2772, 2774, 4005,	\clist_const:Nn 185
4009, 4060	\clist_map_function:nN 3744
\bool_gset_true:N 239, 249, 1062, 1976, 1982, 3981,	\clist_map_inline:Nn 507, 773, 958, 973, 1054, 1499
4006, 4036, 4061	\clist_map_inline:nn . 49, 60, 78, 84, 96, 108, 137,
\bool_if:NTF . 399, 411, 428, 1428, 1505, 1519, 1532,	160, 184, 535, 555, 827, 872, 893, 1068, 1605, 1845,
1543, 1554, 1565, 1576, 1587, 1636, 1653, 1658, 1666,	1912, 2098, 2116, 2148, 2291, 2830, 3090, 3102, 3142,
1693, 1731, 1736, 1743, 1747, 1769, 1774, 1782, 1789,	3252, 3255, 3282, 3294, 3297, 3317, 4748
1820, 1828, 1921, 2119, 2129, 2208, 2232, 2239, 2263,	\columnbreak
2361, 2383, 2423, 2436, 2440, 2490, 2509, 2533, 2587,	\columns 2371
2598, 2687, 2724, 2788, 2821, 2836, 2911, 2922, 2926,	columns
2945, 2958, 3000, 3034, 3069, 3203, 3265, 3275, 3307,	columns-sep
3312, 3415, 3465, 3479, 3487, 3544, 3601, 3614, 3622, 3642, 3977, 3986, 3990, 4032, 4041, 4045, 4134, 4144,	\columnsep
4227, 4232, 4241, 4245, 4260, 4289, 4345, 4447, 4451,	\columnseprule
4475, 4484, 4488, 4494, 4508, 4531	\columnseprule
\bool_if:nTF 1449, 1471, 3056, 3185, 3223, 3663, 4630	Commands provide by enumext:
\bool_if_p:N 258, 273, 831, 832, 842, 843, 1800, 1801,	\anskey 28, 64, 65, 69-73, 75, 76, 83, 84, 94, 109, 118, 119,
1809, 1810, 1934, 1960, 1973, 1974, 1979, 1980, 2296,	126
2306, 2318, 2333, 2334, 2368, 2409, 2410, 2710, 2898,	\anspic* 28, 29, 67, 70, 82-84, 100-102, 118, 119
2899, 2936, 2937, 3388, 3390, 3401, 3670, 3671	\anspic 71, 99-101, 126
\bool_lazy_all:nTF 256, 271, 1932, 1958, 2294, 2303,	\foreachkeyans 122, 128
2316, 2331, 3386, 3399	\getkeyans 70, 118, 129

\item* 28, 29, 67, 70, 71, 82-84, 86, 89, 110, 115, 118, 119	D
\item 86, 89, 104, 109-111, 114, 115	\d 210
\miniright 27, 47, 54, 55, 95, 96, 129	\DeclareDocumentEnvironment 371
\printkeyans* 118	dim commands:
\printkeyans 28, 71, 118, 119	\dim_abs:n 3216, 3221
\setenumextmeta 122, 129	\dim_add:Nn 3722, 3821, 3852
\setenumext 28, 119-121, 124	\dim_compare:nNnTF . 900, 916, 928, 940, 1440, 1464,
Counters defined by enumext:	3213, 3218, 3224, 3230, 3232, 3234, 3425, 3448, 3570,
enumXiii	3588, 3714, 3798, 3814, 3829, 3845, 3958, 4013
enumXii 26, 36	\dim_compare:nTF 2393, 2737, 3344, 3531
enumXiv 26, 36	\dim_gset_eq:NN 3967, 4022
enumXi 26, 36	\dim_gzero:N 2776, 4008, 4063
enumXviii	\dim_new:N . 67, 74, 75, 76, 93, 139, 172, 174, 175, 181
enumXvii	\dim_set:Nn 467, 868, 3032, 3216, 3221, 3223, 3226,
enumXvi 26, 36	3227, 3231, 3233, 3236, 3237, 3239, 3340, 3428, 3451, 3527, 3572, 3590, 3749, 3800, 3807, 3831, 3838, 3893,
enumXv 26, 36	3527, 3572, 3590, 3749, 3600, 3607, 3631, 3636, 3693, 3942, 3960, 4015, 4215
cs commands:	\dim_set_eq:NN 563, 610, 681, 685, 2947, 2948, 2960,
\cs_generate_variant:\text{Nn} \tag{190, 191, 469, 485, 731, 747, 2200, 2205, 2281, 2604, 3242, 3746, 4807}	2961, 3027, 3254, 3296, 3459, 3598, 3900, 3903, 3904,
\cs_if_exist:NTF 439	3949, 3952, 3953, 4206
\cs_if_free:NTF 2555, 2567	\dim_sub:Nn 3349, 3536, 3816, 3847
\cs_new:Nn	\dim_use:N 901, 909, 1441, 1447, 2271, 2274, 2279, 3047,
\cs_new:Npn . 222, 1606, 1615, 1623, 2163, 2172, 2180,	3049, 3346, 3351, 3426, 3431, 3432, 3439, 3449, 3453,
4656, 4665, 4674	3454, 3456
\cs_new_eq:NN 358, 359, 360, 364, 365, 413, 414, 417,	\dim_zero:N 3288, 3463, 3600, 3723, 3724, 3725
418	\dim_zero_new:N 436
\cs_new_protected:Nn . 214, 228, 254, 287, 317, 323,	\c_zero_dim 903, 917, 929, 941, 1441, 1464, 2395, 2739,
329, 335, 341, 349, 367, 385, 596, 659, 711, 828, 974,	3213, 3218, 3224, 3231, 3346, 3426, 3449, 3533, 3570,
978, 982, 986, 990, 994, 998, 1002, 1006, 1010, 1014,	3588, 3798, 3814, 3829, 3845, 3958, 4013
1018, 1022, 1026, 1030, 1034, 1069, 1081, 1105, 1122,	\dimeval 2064
1133, 1150, 1225, 1249, 1266, 1328, 1345, 1367, 1402,	
1408, 1500, 1514, 1528, 1539, 1550, 1561, 1572, 1583,	E
1664, 1767, 1780, 1797, 1818, 1846, 1851, 1876, 1917,	\end 1444, 1467, 2234, 2265, 3478, 3495, 3613, 3630, 3979,
1927, 1970, 1985, 1992, 2001, 2006, 2011, 2016, 2025,	4002, 4034, 4057, 4632, 4641, 4648
2030, 2035, 2206, 2230, 2237, 2261, 2268, 2282, 2507,	\endgroup
2526, 2542, 2605, 2641, 2672, 2707, 2749, 2770, 2778,	\endlist 359
2819, 2834, 2862, 2895, 2931, 2943, 2956, 3042, 3052,	\endlrbox 4279, 4521
3063, 3181, 3197, 3338, 3355, 3384, 3413, 3420, 3443, 3473, 3485, 3525, 3542, 3566, 3584, 3609, 3620, 3659,	\endminipage
3703, 3717, 3742, 3747, 3763, 3767, 3786, 3796, 3827,	enumext internal commands:
3956, 3975, 4011, 4030, 4088, 4116, 4123, 4132, 4142,	\l_enumext_ref_the_count_tl 38
4159, 4300, 4337, 4364, 4370, 4383, 4439, 4543	\l_enumext_resume_name_tl 60
\cs_new_protected:Npn 192, 196, 200, 421, 437, 454,	\enumext_add_meta_key:nnn 4759, 4775, 4776,
464, 470, 576, 621, 693, 718, 732, 1438, 1462, 1632,	4778, 4781
1651, 1721, 1754, 1856, 2040, 2117, 2127, 2149, 2157,	\enumext_add_pre_parsep: . 48, 1079, <u>1081</u> , 1081
2192, 2201, 2357, 2420, 2434, 2472, 2476, 2596, 2627,	\enumext_after_args_exec: . 46,974,986,3331
2631, 2662, 2798, 2872, 2916, 2996, 3015, 3103, 3107,	\enumext_after_args_exec_v: . 990, 1002, 3518
3121, 3125, 3143, 3147, 3157, 3169, 3211, 3245, 3285,	\enumext_after_args_exec_vii: 1006, 1030
3366, 3562, 3712, 3858, 3907, 4105, 4165, 4172, 4188,	\enumext_after_args_exec_viii: 1034
4196, 4201, 4213, 4357, 4389, 4396, 4412, 4420, 4434,	\enumext_after_env:nn . 79, 80, 82, 97, 106, 113,
4564, 4577, 4623, 4745, 4757, 4781, 4793, 4833, 4839,	196, 196, 2682, 3505, 3984, 4039, 4314
4847, 4865	\enumext_after_hyperref: 34, 383, 385, 385
\cs_new_protected_nopar:Nn 4152, 4276, 4376, 4518	\enumext_after_list: . 96, 114, 3336, 3485, 3485
\cs_new_protected_nopar:Npn 4219, 4467	\lenumext_after_list_args_v_tl 1004
\cs_set:Npn 2292, 2329, 4570	\lenumext_after_list_args_vii_tl 1032, 4270
\cs_set_eq:NN 4078, 4079, 4221, 4327, 4328, 4469	\lenumext_after_list_args_viii_tl 1036,
\cs_set_protected:Nn 898, 914, 926, 938	4504
\cs_set_protected:Nn	\enumext_after_list_v: 3523, <u>3566</u> , 3620
156, 164, 486, 508, 540, 556, 603, 748, 774, 818, 854,	\enumext_after_list_vii: 109 , 4086 , 4123 , 4123
877, 950, 959, 1038, 1055, 1483, 1594, 1837, 1898,	\enumext_after_list_viii: 4335, 4370, 4370
2057, 2099, 2135, 2284, 2823, 3079, 3095, 3135, 3243,	\enumext_after_stop_list: 46, 97, <u>974</u> , 982,
3283	3499
\cs_to_str:N 456,479	\enumext_after_stop_list_v: . <u>990</u> , 998, 3637
\cs_undefine:N 2544, 2545, 2546, 2547	\lenumext_after_stop_list_v_tl 1000

132 / 145

\enumext_after_stop_list_vii: 109, 1006,
1022, 4126
\lenumext_after_stop_list_vii_tl 1024
\enumext_after_stop_list_viii: . 1026, 4373
\lenumext_after_stop_list_viii_tl 1028
\lenumext_align_label_vii_str 4262, 4266
\lenumext_align_label_viii_str . 4496, 4500
\lenumext_align_label_X_str <u>164</u>
\cenumext_all_envs_clist 185, 507, 773, 958,
973, 1054, 1499
\cenumext_all_families_seq 121, 4713, 4739
\lenumext_anskey_env_bool 31, 78, <u>34</u> , 264, 279,
2598
\enumext_anskey_env_clean_vars: . 81, 2703,
2707, 2770
\enumext_anskey_env_define_keys: 79, 2596,
2605, 2676
\enumext_anskey_env_exec: 80, 2601, <u>2672</u> , 2672
\enumext_anskey_env_make:n 64, 78, 1881, 2596,
2596, 2604
\enumext_anskey_env_reset_keys: 79, 80, 2641,
2704
\enumext_anskey_env_reset_keys:\
enumext_rescan_anskey_env:n 2596
\enumext_anskey_env_save_keys: 80, 2684,
2707, 2707
\enumext_anskey_env_store: 81, 2700, <u>2707</u> ,
2749
\enumext_anskey_env_unknown:n 79, 2624, 2627
\enumext_anskey_env_unknown:nn . 2629, 2631
\lenumext_anskey_level_int <u>28</u> , 2528, 2529
\enumext_anskey_safe_inner: . 77 , 2501 , 2507 ,
2526
\enumext_anskey_safe_inner:n 76
$\verb \enumext_anskey_safe_outer: . 76, 2488, \underline{2507},$
\enumext_anskey_safe_outer: . 76 , 2488 , $\underline{2507}$, 2507
$\label{eq:content_anskey_safe_outer:} \begin{array}{cccccccccccccccccccccccccccccccccccc$
\enumext_anskey_safe_outer: . 76 , 2488, $\underline{2507}$, 2507 \enumext_anskey_show_wrap_arg:n . 75 , $\underline{2420}$, 2420, 2438, 2453
$\label{eq:content_anskey_safe_outer:} \begin{array}{cccccccccccccccccccccccccccccccccccc$
\enumext_anskey_safe_outer: . 76 , 2488, $\underline{2507}$, 2507 \enumext_anskey_show_wrap_arg:n . 75 , $\underline{2420}$, 2420, 2438, 2453
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434
$\label{eq:content_anskey_safe_outer:} \begin{tabular}{ll} $76,2488,$ $\underline{2507},$ \\ $2507 \\ $$\end{tabular} $$2507 \\ $\end{tabular} $$\end{tabular} $$\end{tabular} $$\end{tabular} $$369,$ $\underline{2420},$ $\underline{2420},$ $\underline{2420},$ $\underline{2420},$ $\underline{2434},$ $\underline{2434},$$
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476
$\label{eq:content_anskey_safe_outer:} \begin{tabular}{ll} $76,2488,$ $\underline{2507},$ \\ $2507 \\ $$\end{tabular} $$2507 \\ $\end{tabular} $$\end{tabular} $$\end{tabular} $$\end{tabular} $$369,$ $\underline{2420},$ $\underline{2420},$ $\underline{2420},$ $\underline{2420},$ $\underline{2434},$ $\underline{2434},$$
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843,
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843, 852
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843,
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843, 852
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843, 852 _enumext_before_args_exec: 46, 95, 108, 974, 974, 3423
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_base_line_fix_bool . 822, 832, 843, 852 \enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_base_line_fix_bool . 822, 832, 843, 852 \enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006,
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_base_line_fix_bool . 822, 832, 843, 852 \enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006, 4120
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_base_line_fix_bool . 822, 832, 843, 852 \enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006,
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_base_line_fix_bool . 822, 832, 843, 852 \enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006, 4120
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 _enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 _enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 _enumext_anskey_unknown:n 76, 2456, 2470, 2472 _enumext_anskey_unknown:nn . 2456, 2474, 2476 _enumext_anskey_wrapper:n 2061, 2432 _enumext_at_begin_document:n 33, 192, 192, 356, 362 \l_enumext_base_line_fix_bool . 822, 832, 843, 852 _enumext_before_args_exec: 46, 95, 108, 974, 974, 3423 _enumext_before_args_exec_v: 990, 990, 3569 _enumext_before_args_exec_vii: . 1006, 1006, 4120 _enumext_before_args_exec_viii: 1010, 4367
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006, 4120 \enumext_before_args_exec_viii: 1010, 4367 \enumext_before_env:nn 79, 196, 200, 2549, 2561, 2573, 2674
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006, 4120 \enumext_before_args_exec_viii: 1010, 4367 _enumext_before_env:nn 79, 196, 200, 2549, 2561, 2573, 2674 \enumext_before_keys_exec: 46, 974, 978, 3328 \enumext_before_keys_exec_v: 990, 994, 3515
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n 75, 2365, 2434, 2434 \enumext_anskey_unknown:n 76, 2456, 2470, 2472 \enumext_anskey_unknown:nn . 2456, 2474, 2476 \enumext_anskey_wrapper:n 2061, 2432 \enumext_at_begin_document:n 33, 192, 192, 356, 362 \lenumext_before_args_exec: 46, 95, 108, 974, 974, 3423 \enumext_before_args_exec_v: 990, 990, 3569 \enumext_before_args_exec_vii: . 1006, 1006, 4120 \enumext_before_args_exec_viii: 1010, 4367 _enumext_before_env:nn 79, 196, 200, 2549, 2561, 2573, 2674 \enumext_before_keys_exec: 46, 974, 978, 3328 \enumext_before_keys_exec_v: 990, 994, 3515
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n
\enumext_anskey_safe_outer: . 76, 2488, 2507, 2507 \enumext_anskey_show_wrap_arg:n . 75, 2420, 2420, 2438, 2453 \enumext_anskey_show_wrap_left:n

```
\__enumext_before_list_viii: . 114, 4319, 4364,
\l__enumext_before_no_starred_key_v_tl
\l__enumext_before_no_starred_key_vii_-
    \l__enumext_before_no_starred_key_viii_-
    \l__enumext_before_starred_key_v_tl ... 992
\l__enumext_before_starred_key_vii_tl . 1008
\l__enumext_before_starred_key_viii_tl 1012
\__enumext_calc_hspace:NNNNNNN 91, 3211, 3211,
   3242, 3247, 3289
\__enumext_check_ans_active: . 65, 95, 108, 1917,
    1917, 3424, 4119
\g__enumext_check_ans_item_tl ..... 84
\g__enumext_check_ans_key_bool 66, 67, 143, 331,
    1976, 1982, 2788
\l__enumext_check_ans_key_bool 66, 1902, 1907,
    1973, 1979
\__enumext_check_ans_key_hook: 66, 97, 109, 1970,
    1970, 3500, 4127
\__enumext_check_ans_level: 65, 1917, 1923, 1927
\__enumext_check_ans_log: 66, 67, 82, 2016, 2016,
    2792
\__enumext_check_ans_log_msg_greater: 2016,
   2022, 2035
\__enumext_check_ans_log_msg_less: 2016, 2020,
\__enumext_check_ans_log_msg_same_ok:
    2021, 2030
\__enumext_check_ans_msg_greater: 1992, 1998,
\__enumext_check_ans_msg_less: 1992, 1996, 2001
\__enumext_check_ans_msg_same_ok: 1992, 1997,
    2006
\__enumext_check_ans_show: . . 66, 82, 1992, 1992,
\l__enumext_check_answers_bool . 64, 65, 76, 86,
    143, 1879, 1906, 1921, 2208, 2232, 2239, 2263, 2490,
    2687, 2911, 3000, 3034, 4232
\__enumext_check_starred_cmd:n 32, 67, 84, 2040,
   2040, 3521, 3698, 4333
\g__enumext_check_starred_cmd_int 143, 2043,
    2049, 2054, 3179, 3668, 4446
\l__enumext_check_start_line_env_tl . 32, 143,
    294, 302, 310, 2046, 2052, 2055
\l__enumext_columns_sep_v_dim 3588, 3590, 3598
\l__enumext_columns_sep_vii_dim . . 3798, 3800,
    3809, 3821, 3897, 4298
\l__enumext_columns_sep_viii_dim . 3829, 3831,
    3840, 3852, 3946, 4541
\l__enumext_columns_v_int 1271, 3586, 3594, 3606,
\l__enumext_columns_vii_int . . 3803, 3806, 3810,
   3819, 3861, 3865, 3868, 3874, 3880, 3884, 4293, 4304
\l__enumext_columns_viii_int . 3834, 3837, 3841,
    3850, 3910, 3914, 3917, 3923, 3929, 3933, 4536, 4549
\l__enumext_counter_i_tl ..... 45, 446
\l__enumext_counter_ii_tl .... 45,447
\l__enumext_counter_iii_tl .... 45,448
\l__enumext_counter_iv_tl .... 45, 449
\c__enumext_counter_style_tl .... 30, 50, 216
\g__enumext_counter_styles_tl . 27, 36, 67, 457,
\l__enumext_counter_v_tl .... 45, 450, 701
```

\lenumext_counter_vi_tl 45, 451
\lenumext_counter_vii_tl 45, 452, 631
\lenumext_counter_viii_tl 45, 453, 648

\lenumext_current_widest_dim $27, \underline{67}, 481, 564,$
611, 682, 686
\enumext_def_meta_key:nnn <u>4759</u> , 4787, 4793,
4807
\enumext_default_item:n 2996, 2996, 3060
\enumext_define_counters:Nn 26, 437, 437, 446,
447, 448, 449, 450, 451, 452, 453
_enumext_endminipage: . 33, 362, 365, 379, 3759,
4278, 4520
\genumext_envir_name_tl 31, 34, 265, 280, 339,
1849, 1854, 1864, 2004, 2009, 2014, 2028, 2033, 2038
$\label{local_local_local_local_local_local} \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$
293, 301, 309, 5149, 5152, 5159, 5162, 5169, 5172,
5179, 5182, 5188, 5192, 5198, 5202, 5259, 5263
\enumext_execute_after_env: 32, 33, 63, 66, 67,
77, 82, 2778, 2778, 3505, 4314
\enumext_fake_item: 898, 898, 3274
\lenumext_fake_item_indent_v_dim 917,922
\lenumext_fake_item_indent_v_tl 919, 3162,
3166, 3174
\lenumext_fake_item_indent_vii_dim 929,934
\lenumext_fake_item_indent_vii_tl 931, 4274
\lenumext_fake_item_indent_viii_dim . 941,
946, 4512
\lenumext_fake_item_indent_viii_tl 943,
4510, 4515
\lenumext_fake_item_indent_X_tl 97
\enumext_fake_item_vii: 898, 926, 3306
\enumext_fake_item_viii: 898, 938, 3311
_enumext_filter_first_level:n 120, 4656,

4656, 4690, 4701
\enumext_filter_first_level_key:n 120, 4656,
4661, 4665
\enumext_filter_first_level_pair:nn . 120,
<u>4656, 4662, 4674</u>
\enumext_filter_save_key:n 70, 2124, 2132,
2155, 2161, 2163, 2163, 4588, 4592, 4596, 4600, 4604,
4608
\enumext_filter_save_key_key:n 70, 2163,
2168, 2172
\enumext_filter_save_key_pair:nn 70, 2163,
2169, 2180
\enumext_filter_series:n 58, <u>1606</u> , 1606, 1644,
1656, 1661
\enumext_filter_series_key:n 58, 1606, 1611,
1615
\enumext_filter_series_pair:nn 59, 1606,
1612 1622
1612, 1623
$\verb \g_enumext_footnote_arg_seq . \underline{161}, 3769, 3782,$
$\label{eq:control_g_enumext_footnote_arg_seq} $$ \frac{161}{3769}, 3782, $$ 3792$
$\label{eq:control_g_seq} $$ \g_{\text{enumext_footnote_int}} . $$ \frac{161}{3769}, 3782, $$ 3792 $$ \g_{\text{enumext_footnote_int}} . $$ \frac{161}{3776}, 3779, 3781, $$ $$$
$\label{eq:control_g_enumext_footnote_arg_seq} $$ \frac{161}{3769}, 3782, $$ 3792$
$\label{eq:control_g_seq} $$ \g_{\text{enumext_footnote_int}} . $$ \frac{161}{3769}, 3782, $$ 3792 $$ \g_{\text{enumext_footnote_int}} . $$ \frac{161}{3776}, 3779, 3781, $$ $$$
$\label{eq:control_g_seq} $$ \g_{\text{enumext_footnote_arg_seq}} . $$ $ \frac{161}{3769}, 3782, $$ 3792 $$ \g_{\text{enumext_footnote_int}} . $$ $ \frac{161}{3776}, 3779, 3781, $$ 3783 $$$
$\label{eq:control_genum} $$ \lg_{\text{enumext_footnote_arg_seq}} . $$ $ \underline{161}, 3769, 3782, $$ $ 3792 $$ $$ g_{\text{enumext_footnote_int}} . $$ \underline{161}, 3776, 3779, 3781, $$ 3783 $$ $$ g_{\text{enumext_footnote_int_seq}} . $$ \underline{161}, 3770, 3783, $$ 3788, 3791 $$$
$\label{eq:control_g_seq} $$ \ \ \ \ \ \frac{161}{3769}, 3782, \\ 3792 $$ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$
$\label{eq:control_g_seq} $$ \ \ \underline{161}, 3769, 3782, 3792 $$ \\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ $
$\label{eq:control_general} $$ \g_{\text{enumext_footnote_arg_seq}} . $$ $ \underline{161}, 3769, 3782, 3792 $$ \\ \g_{\text{enumext_footnote_int}} . $$ \underline{161}, 3776, 3779, 3781, 3783 $$ \\ \g_{\text{enumext_footnote_int_seq}} . $$ \underline{161}, 3770, 3783, 3788, 3791 $$ \\ \g_{\text{enumext_footnotes_key_bool}} $
$\label{eq:controller} $$ \g_{\text{enumext_footnote_arg_seq}} . $$ \underline{161}, 3769, 3782, 3792 $$ \g_{\text{enumext_footnote_int}} . $\underline{161}, 3776, 3779, 3781, 3783 $$ \g_{\text{enumext_footnote_int_seq}} . $\underline{161}, 3770, 3783, 3788, 3791 $$ \g_{\text{enumext_footnotes_key_bool}} $
$\label{eq:controller} $$ \lg_\text{enumext_footnote_arg_seq} . $$ $161, 3769, 3782, 3792 $$ $
$\label{eq:controller} $$ \g_{\text{enumext_footnote_arg_seq}} . $$ \underline{161}, 3769, 3782, 3792 $$ \g_{\text{enumext_footnote_int}} . $\underline{161}, 3776, 3779, 3781, 3783 $$ \g_{\text{enumext_footnote_int_seq}} . $\underline{161}, 3770, 3783, 3788, 3791 $$ \g_{\text{enumext_footnotes_key_bool}} $

```
\g__enumext_foreach_default_keys_tl . . . 123,
    4831, 4849
\__enumext_foreach_kayans:nn .... 4845, 4847
\label{local_enumext_foreach_name_prop_tl} \ \ \ \ _{123,\,4851},
    4872
\l__enumext_foreach_print_seq 123, 4852, 4863,
    4867
\l__enumext_foreach_sep_tl . . . . . . . 4822, 4863
\l__enumext_foreach_start_int .... 4814, 4859
\l__enumext_foreach_step_int .... 4818, 4860
\l__enumext_foreach_stop_int . 4816, 4853, 4855,
\__enumext_foreach_wrapper:n .... 4820, 4870
\__enumext_getkeyans:nn .. 118, 4573, 4577, 4577
\__enumext_getkeyans_aux:n 118, 4561, 4564, 4564
\l__enumext_hyperref_bool . 29, 34, 35, 151, 390,
    411, 428, 2410, 2899, 4227, 4475
\__enumext_if_is_int:n ..... 208
\__enumext_if_is_int:nTF ..... 208, 720, 734
\__enumext_internal_mini_page: 34, 94, 108, 367,
    367, 3357, 4090
\__enumext_is_not_nested: 26, 31, 94, 108, 228, 228,
    3358, 4091
\__enumext_is_on_first_level: . 26, 31, 94, 108,
    <u>228</u>, 254, 3364, 4103
\g__enumext_item_anskey_int 76, 84, 143, 326, 353,
    354, 1989, 2359, 2913
\__enumext_item_answer_diff: .. 66, 67, 82, 1985,
    1985, 2785
\g__enumext_item_answer_diff_int . 66, 67, 143,
    327, 1987, 1994, 2018
\l__enumext_item_column_pos_vii_int 109, 3868,
    3874, 3880, 3884, 3891, 4155, 4293, 4296
\l__enumext_item_column_pos_viii_int .. 114,
    3917, 3923, 3929, 3933, 3940, 4379, 4536, 4539
l__enumext_item_column_pos_X_int .... 164
\g__enumext_item_count_all_vii_int 109, 3892,
    4156, 4304, 4311
\g__enumext_item_count_all_viii_int 114,3941,
    4380, 4548, 4556
\g__enumext_item_count_all_X_int .... 164
\g__enumext_item_number_bool ..... 143
\l__enumext_item_number_bool 65, 149, 1939, 1944,
    1948, 1952, 1965, 2533, 2587, 3003, 3037, 4235
\g__enumext_item_number_int 65, 66, 143, 325, 352,
    354, 1938, 1943, 1947, 1951, 1964, 1989, 3002, 3036,
    4234
\__enumext_item_peek_args_vii: 109, 110, 4157,
    4159, 4159
\__enumext_item_peek_args_viii: .. 114,4381,
    <u>4383</u>, 4383
\__enumext_item_star_exec: . 87, 3015, 3042, 3071
\l__enumext_item_starred_vii_bool 4174,4190,
    4245
\l__enumext_item_starred_viii_bool 4398, 4414,
    4488, 4508
\l__enumext_item_starred_X_bool ..... 164
\__enumext_item_std:w . . 33, 86, 89, 102, 356, 360,
    3006, 3012, 3040, 3162, 3166, 3174, 3735
\g__enumext_item_symbol_aux_tl . 86, 127, 3020,
```

3023, 3048, 3076

4250, 4254, 4256

\g__enumext_item_symbol_aux_vii_tl 4198, 4247,

\g__enumext_item_symbol_aux_X_tl 164

\lenumext_item_symbol_sep_vii_dim 4207,
4215, 4253, 4255
\lenumext_item_symbol_vii_tl 4250
\lenumext_item_text_vii_box 4240, 4281, 4288
\lenumext_item_text_viii_box 4483, 4523, 4530
$\label{local_local_local_local_local} $$ l_enumext_item_text_X_box $$\underline{164}$ $
\lenumext_item_width_vii_dim 3807, 3816,
3895, 3903, 3904
\lenumext_item_width_viii_dim 3838, 3847,
3944, 3952, 3953
\lenumext_item_width_X_dim 164
\lenumext_itemindent_X_dim 71
\lenumext_itemsep_vii_skip 4310
\lenumext_itemsep_viii_skip 4555
\l_enumext_joined_item_aux_vii_int 3889,
3890, 3891, 3892, 3898
\lenumext_joined_item_aux_viii_int . 3938,
3939, 3940, 3941, 3947
\l_enumext_joined_item_aux_X_int 164
\enumext_joined_item_vii:w 110, 4162, 4163,
4165, 4165

\lenumext_joined_item_vii_int 3860, 3861,
3864, 3866, 3872, 3877, 3882, 3887, 3889, 3895
\enumext_joined_item_viii:w 114, 4386, 4387,
<u>4389</u> , 4389
\lenumext_joined_item_viii_int . 3909, 3910,
3913, 3915, 3921, 3926, 3931, 3936, 3938, 3944
$local_loc$
\lenumext_joined_width_vii_dim . 3893, 3900,
3903, 4271, 4283
\lenumext_joined_width_viii_dim 3942, 3949,
3952, 4505, 4525
$\label{local_local_local_local_local} $$ l_enumext_joined_width_X_dim $$ 164$
_enumext_Joined_width_X_dim 104 _enumext_keyans_addto_prop:n 82, 2798, 2798,

\enumext_keyans_addto_prop:n 82 , 2798 , 2798 ,
\enumext_keyans_addto_prop:n 82, <u>2798</u> , <u>2798</u> , <u>2798</u> , <u>3176</u> , <u>3665</u>
\enumext_keyans_addto_prop:n $82, \underline{2798}, 2798, 3176, 3665$ _enumext_keyans_addto_seq:n . $84, \underline{2872}, 2872, 3178, 3667$
$\label{eq:continuous_seq_link} $$ $$ \end{subar} $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$$
\enumext_keyans_addto_prop:n $82, \underline{2798}, 2798, 3176, 3665$ _enumext_keyans_addto_seq:n . $84, \underline{2872}, 2872, 3178, 3667$
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659
$\label{eq:continuous_seq} $$ \ \ $2, \underline{2798}, 2798, \\ 3176, 3665 \\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ $
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193
$\label{eq:continuous_series} $$ \end{align*} $$ \end{align*}$
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: 898, 914, 3264
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \L_enumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: 898, 914, 3264 \L_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n . 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int . 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int . 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n . 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int . 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int . 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266, 1266, 1330
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266, 1266, 1330
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n . 89, 3157, 3157, 3193 \lenumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \l_enumext_keyans_level_h_int . 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \l_enumext_keyans_level_int . 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 _enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266, 1266, 1330 _enumext_keyans_multi_addvspace: 1122, 1133,
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n . 89, 3157, 3157, 3193 \\enumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_fake_item: . 898, 914, 3264 \\enumext_keyans_level_h_int . 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 \\enumext_keyans_level_int . 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 \enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266, 1266, 1330 \enumext_keyans_multi_addvspace: 11122, 1133, 3603
\enumext_keyans_addto_prop:n 82, 2798, 2798, 3176, 3665 \enumext_keyans_addto_seq:n . 84, 2872, 2872, 3178, 3667 \enumext_keyans_addto_seq_link: 2872, 2893, 2895, 4445 \enumext_keyans_anspic_code:nnn 100, 3656, 3659, 3659 \enumext_keyans_default_item:n 89, 3157, 3157, 3193 \\enumext_keyans_env_bool 34, 3389, 3402, 3549, 3636 \enumext_keyans_level_h_int 28, 641, 668, 2517, 2579, 2850, 4097, 4339, 4340 _enumext_keyans_level_int 28, 1432, 2513, 2575, 2845, 3548, 3553, 3650 \enumext_keyans_make_label: 37, 90, 3181, 3197, 3262 _enumext_keyans_mini_addvspace: . 53, 1328, 1328, 3578 _enumext_keyans_mini_right_cmd:n 55, 1434, 1462, 1462 _enumext_keyans_mini_set_vskip: . 52, 1266, 1266, 1330 _enumext_keyans_multi_addvspace: 1122, 1133, 3603 _enumext_keyans_multi_set_vskip: 49, 1122,

```
3609, 3634
\__enumext_keyans_name_and_start: 26, 32, 287,
    287, 3550, 3710, 4344
\__enumext_keyans_parse_keys:n 3509, 3562, 3562
\l__enumext_keyans_pic_above_int . 138, 3750,
    3751, 3753
\l__enumext_keyans_pic_above_skip . 102, 138,
    3689, 3729
\__enumext_keyans_pic_arg_two: 101, 3687, 3717,
\l__enumext_keyans_pic_below_int . 138, 3750,
    3751, 3754
\l__enumext_keyans_pic_body_seq 100-102, 138,
    3654, 3694, 3758
\__enumext_keyans_pic_do:n 102, 3694, 3696, 3742,
    3742, 3746
\l__enumext_keyans_pic_level_int .. <u>28</u>, 1416,
    2521, 2583, 2801, 2840, 2875, 2963, 3705, 3706
\__enumext_keyans_pic_row:n . . . 102, 3744, 3747,
\__enumext_keyans_pic_safe_exec: . 101, 3683,
    3703, 3703
\__enumext_keyans_pic_skip_abs:N . 101, 3712,
    3712, 3728
\l__enumext_keyans_pic_width_dim . 138, 3749,
    3756
\__enumext_keyans_redefine_item: .. 90, 3181,
    3181, 3261
\__enumext_keyans_ref: .... 41,693,711,3263
\__enumext_keyans_ref:n .... 40,690,693,693
\__enumext_keyans_safe_exec: . 3508, 3542, 3542
\__enumext_keyans_set_item_width: . 97,3517,
    3525, 3525
\__enumext_keyans_show_ans: . . 2916, 2924, 2943
\__enumext_keyans_show_item_opt: . 2916, 2931,
    3174, 3679, 4511
\__enumext_keyans_show_left:n . 89, 2916, 2916,
    3172, 3674
\__enumext_keyans_show_pos: . . 2916, 2928, 2956
\__enumext_keyans_starred_item:n .. 89, 3169,
    3169, 3189
\__enumext_keyans_store_ref: . . 83, 2819, 2819,
    3177, 3666, 4443
\__enumext_keyans_store_ref_aux_i:
                                       83, 2819,
    2831, 2834
\__enumext_keyans_store_ref_aux_ii: 83, 2819,
    2860, 2862
\__enumext_keyans_unknown_keys:n . 3095, 3099,
\__enumext_keyans_unknown_keys:nn 3095, 3105,
\__enumext_keyans_wrapper_opt:n . . 2067, 2939
\l__enumext_label_copy_i_tl . . 2325, 2838, 2843,
    2848, 2853
\l__enumext_label_copy_v_tl .... 2848
\l__enumext_label_copy_vi_tl ..... 2843
\l__enumext_label_copy_vii_tl 2301, 2312, 2341,
\l__enumext_label_copy_viii_tl .... 2853
\l__enumext_label_copy_X_tl ..... 153
\l__enumext_label_fill_left_v_tl .... 3201
\l__enumext_label_fill_left_X_tl ..... 97
\l__enumext_label_fill_right_v_tl .... 3208
```

__enumext_keyans_multicols_stop: 1466, 3566,

\lenumext_label_fill_right_X_tl 97
\lenumext_label_font_style_v_tl 3202, 3678
\lenumext_label_font_style_vii_tl 4259
\lenumext_label_font_style_viii_tl 4493
\lenumext_label_i_tl 556
\lenumext_label_ii_tl 556
\lenumext_label_iii_tl 556
\lenumext_label_iv_tl 556
\enumext_label_style:Nnn 26, 36, <u>470</u> , 470, 485,
561, 608, 679, 683
\lenumext_label_v_tl 82, 84, 676, 2806, 2880,
2950, 2990, 3171, 3175, 3512, 3673, 3675
\lenumext_label_vi_tl . 82, 84, 676, 2803, 2877,
3673, 3675, 3679
\label_{vii_tl} . $\underline{603}$, 4185 , 4210 , 4217
$\label_{viii_tl} = 603, 4409, 4437, 4441$
$\label_width_by_box 67, 466, 467$
$\verb \colored= width_by_box:Nn 36, \underline{464}, \underline{464},$
469, 481, 744
\lenumext_labelsep_i_dim 2948, 2953, 2961,
2993, 4449, 4464
\l_enumext_labelsep_v_dim 3593
\lenumext_labelsep_vii_dim . 2425, 2948, 2961,
3802, 3812, 3896, 4208, 4269, 4285
\lenumext_labelsep_viii_dim 3833, 3843, 3945,
4503, 4512, 4527
\lenumext_labelwidth_i_dim . 2947, 2953, 2960,
2993, 4449, 4464
2993, 4449, 4404 \lenumext_labelwidth_v_dim 3593
\lenumext_labelwidth_vii_dim 2425, 2947,
2960, 3802, 3811, 3896, 4262, 4266, 4284
\lenumext_labelwidth_viii_dim 3833, 3842,
3945, 4496, 4500, 4526
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . <i>101</i> , 3719
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool <u>71</u>
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . <i>101</i> , 3719
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool <u>71</u>
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool 71 \l_enumext_leftmargin_tmp_X_dim 71 \l_enumext_leftmargin_X_dim 71 _enumext_level: 204, 204, 585, 588, 589, 598, 600, 901, 905, 909, 976, 980, 984, 988, 1071, 1073, 1075,
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool 71 \l_enumext_leftmargin_tmp_X_dim 71 \l_enumext_leftmargin_X_dim 71 \l_enumext_leftmargin_X_dim 71 _enumext_level: 204, 204, 585, 588, 589, 598, 600, 901, 905, 909, 976, 980, 984, 988, 1071, 1073, 1075, 1077, 1110, 1112, 1114, 1116, 1120, 1153, 1156, 1175, 1184, 1190, 1195, 1199, 1210, 1214, 1215, 1220, 1256, 1260, 1441, 1447, 1503, 1505, 1507, 1510, 1517, 1519, 1521, 1524, 2119, 2121, 2123, 2151, 2152, 2154, 2210, 2218, 2222, 2226, 2429, 2430, 3005, 3006, 3010, 3011,
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool
3945, 4496, 4500, 4526 \lenumext_leftmargin_tmp_v_bool . 101, 3719 \lenumext_leftmargin_tmp_X_bool

```
\l__enumext_listoffset_v_dim . 3533, 3538, 3595
\l__enumext_listparindent_vii_dim .... 4272
\l__enumext_listparindent_viii_dim ... 4506
\__enumext_log_answer_vars: . 33, 341, 349, 2787
\__enumext_log_global_vars: . 33, 341, 341, 2786
\__enumext_make_label: .... 37, 87, 3063, 3272
\l__enumext_mark_answer_sym_tl 72, 2073, 2276,
    2442, 2965, 2978, 4453
\l__enumext_mark_position_str 127, 2077, 2078,
    2104, 2105, 2274
\l__enumext_mark_ref_sym_tl . . 2090, 2415, 2907
\l__enumext_meta_path_tl . 123, 4783, 4784, 4786,
    4787
\c__enumext_meta_paths_prop . . . . . 4759, 4786
\__enumext_mini_addvspace: . . 51, 95, 1249, 1249,
\__enumext_mini_addvspace_vii: 54, 1402, 1402,
    3970
\__enumext_mini_addvspace_viii: 54, 1402, 1408,
    4025
__enumext_mini_env* ..... 367
\__enumext_mini_right_cmd:n 55, 1436, 1438, 1438
\__enumext_mini_set_vskip: . 50, 1150, 1150, 1251
\__enumext_mini_set_vskip_vii: 53, 1345, 1345,
    1404
\__enumext_mini_set_vskip_viii: 53, 1345, 1367,
    1410
\__enumext_minipage:w 33, 362, 364, 373, 3756, 4271,
    4505
\l__enumext_minipage_active_v_bool 3576, 3601,
    3614, 3622
\g__enumext_minipage_active_vii_bool . . 106,
    3981, 3986, 4005
\l__enumext_minipage_active_vii_bool . 3966,
\g__enumext_minipage_active_viii_bool
                                         4036,
    4041, 4060
\l__enumext_minipage_active_viii_bool 4021,
\g__enumext_minipage_active_X_bool ... 164
\l__enumext_minipage_active_X_bool .... 85
\g__enumext_minipage_after_skip 85, 1349, 1361,
    4003, 4058
\l__enumext_minipage_after_skip 50, 51, 96, 85,
    1166, 1181, 1201, 1217, 1232, 1238, 1244, 1258, 1268,
    1277, 1280, 1292, 1310, 1321, 1337, 1369, 1382, 1396,
    3496, 3631
\g__enumext_minipage_center_vii_bool . 3990,
\g__enumext_minipage_center_viii_bool 4045,
    4061
\g__enumext_minipage_center_X_bool ... 164
\l__enumext_minipage_hsep_v_dim .... 3574
\l__enumext_minipage_hsep_vii_dim .... 3964
\l__enumext_minipage_hsep_viii_dim ... 4019
\l__enumext_minipage_left_skip . . 50, 85, 1158,
    1173, 1192, 1207, 1254, 1264, 1269, 1275, 1284, 1301,
    1313, 1333, 1343, 1347, 1352, 1356, 1370, 1374, 1388,
    1406, 1412
\l__enumext_minipage_left_v_dim .. 3572, 3580
\l__enumext_minipage_left_vii_dim 3960, 3972
\l__enumext_minipage_left_viii_dim 4015, 4027
\l__enumext_minipage_left_X_dim ..... 85
```

\g__enumext_minipage_right_skip 85, 1348, 1353, 1357, 3989, 4044 \l__enumext_minipage_right_skip . 50, 85, 1162, 1177, 1197, 1212, 1270, 1276, 1288, 1306, 1317, 1371, 1378, 1392, 1448, 1470 \l__enumext_minipage_right_v_dim . 1464, 1469, 3570, 3574 \g__enumext_minipage_right_vii_dim 106, 3968, 3988, 4008 \l__enumext_minipage_right_vii_dim 106, 3958, 3963, 3969 \g__enumext_minipage_right_viii_dim . . 4023, \l__enumext_minipage_right_viii_dim . . 4013, 4018, 4024 \g__enumext_minipage_right_X_dim 164 \g__enumext_minipage_right_X_skip 164 \g__enumext_minipage_stat_int 95, 85, 1453, 1475, 3435, 3489, 3494, 3577, 3624, 3629 \l__enumext_miniright_code_vii_box 3997, 4001 \g__enumext_miniright_code_vii_tl 106, 3992, 3999, 4007 \l__enumext_miniright_code_viii_box . . 4052, \g__enumext_miniright_code_viii_tl 4047, 4054, \l__enumext_miniright_code_X_box 164 __enumext_multi_addvspace: . 49, 96, 1105, 1105, __enumext_multi_set_vskip: 48, 1069, 1069, 1107 \l__enumext_multicols_above_ii_skip . . . 1088 \l__enumext_multicols_above_iii_skip . . 1094 \l__enumext_multicols_above_iv_skip ... 1100 \l__enumext_multicols_above_v_skip 1124, 1138, \l__enumext_multicols_above_X_skip 79 \l__enumext_multicols_below_v_skip 1128, 1142, \l__enumext_multicols_below_X_skip 79 __enumext_multicols_start: . 95, 96, 3441, 3443, __enumext_multicols_stop: 96, 1443, 3473, 3473, 3498 __enumext_nested_base_line_fix: . 43, 94, 108, <u>818</u>, 828, 3375, 4113 __enumext_newlabel:nn 29, 35, 73, 421, 421, 2351, \l_enumext_newlabel_arg_one_tl 29, 35, 73, 83, 153, 2344, 2352, 2414, 2855, 2867, 2905 \l_enumext_newlabel_arg_two_tl 29, 35, 72, 153, 2300, 2310, 2323, 2338, 2353, 2842, 2847, 2852, 2868 __enumext_parse_foreach_keys:n . . 4824, 4839 $\ensuremath{\mbox{\sc loss}}$ enumext_parse_foreach_keys:nn . $4833,\,4841$ __enumext_parse_keys:n 43, 59, 3321, 3366, 3366 __enumext_parse_keys_vii:n . 43, 59, 4068, 4105, __enumext_parse_keys_viii:n . 4318, 4357, 4357 __enumext_parse_save_key:n 69, 2144, 2149, 2149 __enumext_parse_save_key_vii:n 69, 2139, 2149, __enumext_parse_series:n 59, 94, 108, 1632, 1632, __enumext_parse_store_keys:n 94 \l__enumext_parsep_i_skip 1086, 1088, 1230, 1278 \l__enumext_parsep_ii_skip . . . 1092, 1094, 1236

```
\l__enumext_parsep_iii_skip . . 1098, 1100, 1242
\l__enumext_parsep_vii_skip ..... 4273
\l__enumext_parsep_viii_skip ..... 4507
\l__enumext_partopsep_v_skip . 1140, 1144, 1304,
    1308, 1315, 1319, 1335, 1339
\l__enumext_partopsep_viii_skip ..... 1380
\__enumext_phantomsection: 35, 385, 414, 418, 434
\__enumext_print_footnote: ... 3763, 3786, 4291,
    4533
\__enumext_print_keyans_box:NN 72, 2268, 2268,
    2281, 2425, 2428, 2952, 2992, 4449, 4464
\l__enumext_print_keyans_i_tl .... 4593, 4615
\l__enumext_print_keyans_ii_tl ... 4597, 4616
\l__enumext_print_keyans_iii_tl .. 4601, 4617
\l__enumext_print_keyans_iv_tl ... 4605, 4618
\l__enumext_print_keyans_starred_tl 118, 119,
    127, 4589, 4637
\l__enumext_print_keyans_vii_tl 118, 4609, 4619
\l__enumext_print_keyans_X_tl ..... 127
\__enumext_printkeyans:nnn 119, 4620, 4623, 4623
\__enumext_redefine_item: . 87, 3052, 3052, 3271
\l__enumext_ref_key_arg_tl 38, 50, 219, 578, 579,
    592, 623, 626, 637, 643, 654, 695, 696, 707
\label{local_enumext_ref_the_count_tl} 1. 38, 50, 585, 588,
    591, 631, 633, 636, 648, 650, 653, 701, 703, 706
\__enumext_regex_counter_style: . . 30, 38, 214,
    214, 586, 632, 649, 702
\__enumext_register_counter_style:Nn .. 454,
    454, 459, 460, 461, 462, 463
\__enumext_remove_extra_parsep_vii: .. 4083,
    4300, 4300
\__enumext_remove_extra_parsep_viii: . 4332,
    <u>4543</u>, 4543
\__enumext_renew_footnote: . . . 3763, 3767, 4243,
    4486
\l__enumext_renew_the_count_v_tl 704,713,715
\l__enumext_renew_the_count_vii_tl 634,663,
\l__enumext_renew_the_count_viii_tl 651,670,
\l__enumext_renew_the_count_X_tl ..... 50
\__enumext_rescan_anskey_env:n . . 79, 81, 2662,
    2757, 2765
\__enumext_reset_global_bool: .. 317, 320, 329
\__enumext_reset_global_int: ... 317, 319, 323
\__enumext_reset_global_tl: .... 317, 321, 335
\__enumext_reset_global_vars: . 32, 82, 317, 317,
l_enumext_resume_active_bool 59, 61, 61, 1636,
\__enumext_resume_counter: . 61, 1754, 1760, 1767
\__enumext_resume_counter:n . 59, 61, 1725, 1730,
    1754, 1754, 1824, 1832
\__enumext_resume_counter_save_ans: . . 61, 62,
    <u>1754</u>, 1765, 1797
\__enumext_resume_counter_series: 61, 62, 1754,
    1763, 1780
\g_{\text{enumext\_resume\_int}} . . . \underline{61}, 1677, 1771, 1772
\__enumext_resume_last:n . . 59, 1632, 1638, 1651
\l__enumext_resume_name_tl 61, 1673, 1681, 1684,
    1700, 1708, 1711, 1757, 1758, 1786, 1793
\__enumext_resume_save_counter: .. 60, 97, 109,
    1664, 1664, 3503, 4130
```

__enumext_resume_series:n . 61, 1600, 1721, 1721

__enumext_resume_starred: . 62, 1601, <u>1818</u>, 1818 \g__enumext_resume_vii_int <u>61</u>, 1704, 1776, 1777 \l__enumext_rightmargin_vii_dim .. 3814, 3818, \l__enumext_rightmargin_viii_dim . 3845, 3849, 3854 $\ensuremath{\mbox{\mbox{--}enumext_safe_exec:}}\ \dots\ 34,\ 94,\ 3320,\ \underline{3355},\ 3355$ __enumext_safe_exec_vii: . 34, 4067, 4088, 4088 __enumext_safe_exec_viii: . . . 4317, 4337, 4337 \l__enumext_series_name_tl 61 \l__enumext_series_str .. 60, 94, 108, 1598, 1634, 1642, 1643, 1645, 1647, 1668, 1671, 1675, 1695, 1698, 1702, 3370, 4109 __enumext_set_error:nn 4745, 4755, 4757 __enumext_set_item_width: . 93, 3330, 3338, 3338 $\verb|__enumext_set_parse:n 4729, \underline{4745}, \underline{4745}, \underline{4745}$ \l__enumext_setkey_tmpa_int . . . <u>118</u>, 4722, 4726 $\verb|\lower| \verb| l_enumext_setkey_tmpa_seq ... | \underline{118}, 4720, 4730,$ 4736, 4738, 4740, 4752 \l__enumext_setkey_tmpa_tl 118, 4728, 4732 $\label{local_loc$ 4728, 4729 \l__enumext_setkey_tmpb_tl <u>118</u>, 4747, 4749, 4750 \l__enumext_show_answer_bool . 2084, 2108, 2436, 2922, 2936, 3670, 4447 __enumext_show_length:nnn . . 45, 222, 222, 4960, 4961, 4962, 4963, 4964, 4965, 4966, 4967, 4968, 4969, 4975, 4976, 4977, 4978, 4979, 4980, 4981, 4982, 4983, 4984 \l__enumext_show_position_bool ... 2087, 2111, 2440, 2926, 2937, 3671, 4451 \g__enumext_standar_bool 31, 94, 34, 236, 239, 258, $332,\,1666,\,1731,\,1743,\,1769,\,1782,\,1820,\,1960,\,1974,$ 2305, 2318, 2333, 3390 \l__enumext_standar_bool . 94, 97, 34, 2306, 3362, 3502, 4102 \l__enumext_standar_first_bool 31, 94, 34, 263, 831, 1653, 1800, 1862, 1869 __enumext_standar_item_vii:w . 110, 4170, 4172, 4172 __enumext_standar_item_viii:w 114, 115, 4394, 4396, 4396 __enumext_standar_ref: ... 39, 576, 596, 3273 __enumext_standar_ref:n 38, 568, 576, 576 \g__enumext_standar_series_tl . 61, 1655, 1656, 1822, 1825 __enumext_standar_unknown_keys:n 3135, 3139, __enumext_standar_unknown_keys:nn 3135, 3145, \g__enumext_starred_bool *31*, *108*, *34*, 246, 249, 273, $333,\,1693,\,1736,\,1747,\,1774,\,1789,\,1828,\,1934,\,1980,$ 2296, 2836, 4009 \l__enumext_starred_bool 108, 109, 34, 1428, 2334, 2369, 2375, 2423, 2711, 2716, 2945, 2958, 3363, 4101, 4129, 4345, 4349 __enumext_starred_columns_set_vii: .. 3796, 3796, 4076 __enumext_starred_columns_set_viii: . 3796, \l__enumext_starred_first_bool 31, 108, 34, 278, 842, 1658, 1809, 1862, 1869 __enumext_starred_item:nn . . . 3015, 3015, 3058

__enumext_starred_item_exec: . 115, 4439, 4439,

```
\__enumext_starred_item_vii:w . 110, 4169, 4188,
    4188
\__enumext_starred_item_vii_aux_i:w . . 4188,
    4193, 4196
\__enumext_starred_item_vii_aux_ii:w . 4188,
    4194, 4199, 4201
\__enumext_starred_item_vii_aux_iii:w 4188,
    4204, 4213
\__enumext_starred_item_viii:w 114, 115, 4393,
    4412, 4412
\__enumext_starred_item_viii_aux_i:w .. 115,
    4412, 4417, 4420
\__enumext_starred_item_viii_aux_ii:w . 115,
    4412, 4418, 4432, 4434
\__enumext_starred_joined_item_vii:n 104, 110,
    3858, 3858, 4167
\__enumext_starred_joined_item_viii:n . 104,
    114, 3858, 3907, 4391
\__enumext_starred_ref: \dots 40, 621, 659, 3303
\__enumext_starred_ref:n ... 39, 615, 621, 621
\g__enumext_starred_series_tl . <u>61</u>, 1660, 1661,
    1830, 1833
\__enumext_starred_unknown_keys:n 3117, 3119,
    3121
\__enumext_starred_unknown_keys:nn 3117, 3123,
\__enumext_start_from:NNn 41,718,718,731,753,
\l__enumext_start_i_int .... 1772, 1784, 1803
\__enumext_start_item_tmp_vii: 107, 4079, 4152,
    4152
\__enumext_start_item_tmp_viii: .. 113, 4328,
    4376, 4376
\__enumext_start_item_vii:w 110, 111, 4180, 4185,
    4210, 4217, 4219, 4219
\__enumext_start_item_viii:w . . 115, 4404, 4409,
    4437, 4467, 4467
\g__enumext_start_line_tl 31, 34, 266, 281, 338,
    2004, 2009, 2014, 2028, 2033, 2038
\__enumext_start_list:nn . . 33, 90, 101, 356, 358,
    3324, 3511, 3684, 4071, 4320
\__enumext_start_mini_vii: 108, 3956, 3956, 4121
\__enumext_start_mini_viii: . . 114, 4011, 4011,
    4368
\__enumext_start_save_ans_msg: 63, 1846, 1846,
    1871
\__enumext_start_store_level: . 94, 3323, <u>3384</u>,
    3384
\__enumext_start_store_level_vii:
                                        109, 4070,
    4132, 4132
\l__enumext_start_vii_int ... 1777, 1791, 1812
\l__enumext_start_X_int ...... 97
\__enumext_stop_item_tmp_vii: . . 107, 109, 111,
    4078, 4082, 4154, 4221
\__enumext_stop_item_tmp_viii: 113, 114, 4327,
    4331, 4378, 4469
\__enumext_stop_item_vii: 111, 112, 4221, 4276,
\__enumext_stop_item_viii: 117, 4469, 4518, 4518
\__enumext_stop_list: . . 33, <u>356</u>, 359, 3334, 3522,
    3697, 4084, 4334
\__enumext_stop_mini_vii:
                               106, 109, 39<u>56,</u> 3975,
    4125
```

- __enumext_stop_mini_viii: 114, 4011, 4030, 4372 __enumext_stop_save_ans_msg: . 63, 1846, 1851, 2784 __enumext_stop_store_level: . . 94, 3335, 3384, __enumext_stop_store_level_vii: . 109, 4085, 4132, 4142 \l__enumext_store_active_bool 28, 64, 109, 1801, 1810, 1878, 2509, 3388, 3401, 3544, 3552, 3642, 3701, 4134, 4144, 4351 __enumext_store_active_keys:n .. 69, 94, 2117, 2117, 3381 __enumext_store_active_keys_vii:n . 69, 108, 2117, 2127, 4112 __enumext_store_addto_prop:n 70, 82, 2192, 2192, 2200, 2360, 2817, 4442 __enumext_store_addto_seq:n 71, 84, 2201, 2201, 2205, 2212, 2226, 2234, 2243, 2257, 2265, 2418, 2910 \l__enumext_store_anskey_arg_tl .. 28, 74, 109, 2366, 2371, 2373, 2378, 2385, 2388, 2398, 2403, 2406, 2412, 2418 __enumext_store_anskey_code:n 74, 76, 81, 2357, 2357, 2502, 2755, 2763 \l__enumext_store_anskey_env_tl .. 28, 80, 109, 2685, 2689, 2695, 2757, 2765 \l__enumext_store_anskey_opt_tl 28, 80, 81, 109, 2686, 2713, 2719, 2726, 2732, 2742, 2752, 2761 __enumext_store_anskey_safe_outer: 77 \g__enumext_store_columns_break_bool . 2609, 2710, 2772 \l__enumext_store_columns_break_bool . 2368, \l__enumext_store_current_label_tl 28, 82, 84, 115, 109, 2800, 2803, 2806, 2813, 2815, 2817, 2874, 2877, 2880, 2886, 2891, 2901, 2910, 4422, 4427, 4428, 4441, 4442, 4444 \l__enumext_store_current_label_tmp_tl . 28, 109, 3171, 3175 \l__enumext_store_current_opt_arg_tl 28, 115, 109, 2920, 2933, 2939, 4430 __enumext_store_internal_ref: .. 72, 74, 2282, \g__enumext_store_item_join_int .. 2612, 2717, 2721, 2773 \l__enumext_store_item_join_int .. 2376, 2380, \g__enumext_store_item_star_bool . 2614, 2724, \l__enumext_store_item_star_bool . 2383, 2463 \g__enumext_store_item_symbol_sep_dim 2619, 2739, 2744, 2776 \l__enumext_store_item_symbol_sep_dim 2395, 2400, 2468 \g__enumext_store_item_symbol_tl . 2617, 2730, 2734, 2775 \l__enumext_store_item_symbol_tl . 2386, 2390, 2466 \l__enumext_store_keyans_item_opt_sep_tl 2070, 2811, 2813, 2884, 2888, 4425, 4427 __enumext_store_level_close: . 71, <u>2206</u>, 2230, 3417 __enumext_store_level_close_vii: . 71, 2237, __enumext_store_level_open: 71, 95, 2206, 2206,
- 2237, 4138 \g__enumext_store_name_tl 28, 64, 109, 337, 344, 345, 346, 347, 1854, 1880, 2003, 2008, 2013, 2027, 2032, 2037, 2782 \l__enumext_store_name_tl 28, 63, 65, 109, 1687, 1690, 1714, 1717, 1805, 1814, 1849, 1858, 1859, 1880, 1881, 1882, 1884, 1885, 1887, 1889, 1890, 1892, 1894, 1895, 1919, 2194, 2196, 2203, 2346, 2347, 2448, 2691, 2857, 2858, 2971, 2984, 4459 \l__enumext_store_ref_key_bool 74, 2093, 2361, 2409, 2821, 2898 \l__enumext_store_save_key_vii_bool . . 2129, \l__enumext_store_save_key_vii_tl 2131, 2132, 2160, 2161, 2241, 2249, 2253, 2257 \l__enumext_store_save_key_X_bool .. 69, 127 \l__enumext_store_save_key_X_tl 69, 127 \l__enumext_store_upper_level_X_bool . . 127 __enumext_storing_exec: . 63, 64, 78, 1856, 1872, __enumext_storing_set:n . . 63, 1841, 1856, 1856 \l__enumext_the_counter_v_tl 703 \l__enumext_the_counter_vii_tl 633 \l__enumext_the_counter_viii_tl 650 \l__enumext_the_counter_X_tl 50 __enumext_tmp:n 45, 49, 54, 60, 71, 78, 79, 84, 91, 96, 97, 108, 130, 137, 156, 160, 164, 184, 818, 827, 1594, 1605, 1837, 1845, 1898, 1916, 2057, 2098, 2099, 2116, 2135, 2148, 2284, 2291, 2292, 2313, 2326, 2329, 2340, 2823, 2830, 3095, 3102, 3135, 3142, 3243, 3282, 3283, 3317 __enumext_tmp:nn 486, 507, 508, 539, 540, 555, 748, 773, 854, 876, 877, 897, 950, 958, 959, 973, 1038, 1054, 1055, 1068, 1483, 1499, 3079, 3094 __enumext_tmp:nnn 556, 572, 573, 574, 575, 603, 619, __enumext_tmp:nnnnnn 774,799,802,805,807,809, 812, 815 __enumext_tmp:w 4570, 4573 \l__enumext_tmpa_vii_int 3806, 3809, 3818, 3849 \l__enumext_tmpa_viii_int 3837, 3840 \l__enumext_tmpa_X_dim 164 \l__enumext_tmpa_X_int 164 \l__enumext_topsep_v_skip 1126, 1130, 1273, 1286, 1294, 1299, 1319, 1323, 3700, 3732 \l__enumext_topsep_vii_skip . . 1350, 1359, 1363 \l__enumext_topsep_viii_skip . 1372, 1394, 1398 __enumext_undefine_anskey_env: . 77, 82, 2542, 2542, 2793 \l__enumext_vspace_a_star_v_bool 1532 \l__enumext_vspace_a_star_vii_bool ... 1554 \l__enumext_vspace_a_star_viii_bool . . . 1565 \l__enumext_vspace_a_star_X_bool 97 __enumext_vspace_above: 56, 95, 1500, 1500, 3422 __enumext_vspace_above_v: . 57, 1528, 1528, 3568 \l__enumext_vspace_above_v_skip .. 1530, 1534, __enumext_vspace_above_vii: 57, 108, 1550, 1550, 4118 \l__enumext_vspace_above_vii_skip 1552, 1556, __enumext_vspace_above_viii: . 57, 1550, 1561, 4366

__enumext_store_level_open_vii: .. 71, 2237,

3396, 3409

\lenumext_vspace_above_viii_skip 1563, 1567,	106, 114, 125, 127
1569	Environments:
\lenumext_vspace_b_star_v_bool 1543	list 30, 33, 90, 91, 9
\lenumext_vspace_b_star_vii_bool 1576	lrbox 103, 111, 112, 116, 11
\lenumext_vspace_b_star_viii_bool 1587	minipage . 30, 33, 34, 47, 49, 50, 99–103, 111, 112, 11
\lenumext_vspace_b_star_X_bool <u>97</u>	multicols
\enumext_vspace_below: 56, 97, 1514, 1514, 3501	scontents
\enumext_vspace_below_v: . 57, 1539, 1539, 3638	exp commands:
\lenumext_vspace_below_v_skip 1541, 1545,	\exp_after:wN 457
1547	\exp_args:Ne 2754, 2762, 3378, 456
\enumext_vspace_below_vii: 58, 109, 1572, 1572,	\exp_args:NV 2474, 2629, 3105, 3123, 3145, 484
4128	\exp_not:N . 58, 477, 591, 636, 653, 706, 907, 921, 922
\l_enumext_vspace_below_vii_skip 1574, 1578,	933, 934, 945, 946, 2414, 2445, 2446, 2903, 2968, 2969
1580	2981, 2982, 4456, 4457, 4570
\enumext_vspace_below_viii: . 58, 1572, 1583,	\exp_not:n 268, 283, 296, 304, 312, 530, 550, 591, 592
4374	636, 637, 653, 654, 706, 707, 908, 1621, 1630, 2081,
\lenumext_vspace_below_viii_skip 1585,1589,	2178, 2190, 2352, 2380, 2390, 2400, 2414, 2415, 2721
	2734, 2744, 2867, 2905, 2907, 4672, 4682, 4869, 4874
1591 \enumext_widest_from:nNNn 41,732,732,747,	-/31, -/11,/, -/-3, -/-/, 1-/-, 1, 1/, 1-/-
	F
766	\fbox 206
\genumext_widest_label_tl $27, 36, \underline{67}, 474, 478,$	\fboxrule 200
482	•
\lenumext_wrap_label_opt_v_bool 3165	\fboxsep 206
<pre>\lenumext_wrap_label_opt_vii_bool 110, 4179</pre>	file commands:
<pre>\lenumext_wrap_label_opt_viii_bool 115,</pre>	\file_input_stop: 527
4403	first <u>95</u>
\lenumext_wrap_label_opt_X_bool <u>97</u>	font
\lenumext_wrap_label_v_bool 3161, 3165, 3173,	\footnote 10
3203	\footnote 102, 377
\lenumext_wrap_label_vii_bool 110,4178,	\footnotemark 378
4183, 4191, 4260	\footnotesize 2446, 2969, 2982, 445
\lenumext_wrap_label_viii_bool . 115, 4402,	\footnotetext 376
4407, 4415, 4494	\foreachkayans 484
<pre>\lenumext_wrap_label_X_bool 97</pre>	\foreachkeyans 12
\enumext_wrapper_label_v:n 3205, 3679	
\enumext_wrapper_label_vii:n 4263	G
\enumext_wrapper_label_viii:n 4497	\getkeyans 16, 118, 455
\lenumext_write_aux_file_tl . 29, 73, 83, 153,	group commands:
2349, 2355, 2864, 2870	\group_begin: 2444, 2489, 2664, 2751, 2967, 2986
\enumext_zero_parsep: 51, 1170, 1225, 1225	4239, 4258, 4455, 4482, 4492, 4581, 4614
enumext*	\group_end: 2451, 2505, 2768, 2974, 2987, 4268, 4286
- <u> </u>	4462, 4502, 4522, 4583, 4621
<u></u>	11 715 715 715 571
<u> </u>	Н
enumXiii	\hbadness 4287, 452
enumXiv	hbox commands:
enumXv	\hbox_set:Nn 46
enumXvi <u>446</u>	\hfill 516, 520, 525, 526, 1445, 1468, 2414, 2903, 3980, 403
enumXvii <u>446</u>	hook commands:
enumXviii <u>446</u>	
Environments provide by enumext:	\hook_gput_code:nnn 9, 194, 198, 202, 38
anskey* 28, 64, 72, 73, 75, 77, 78, 80, 82, 94, 109, 119, 124,	\hook_gremove_code:nn 80, 268
126	\hook_gset_rule:nnnn 38
enumext* 25, 26, 29-31, 34, 36, 39-45, 47, 53, 54, 57-63,	\hook_if_empty:nTF 267
65, 66, 68–78, 80–83, 88, 92, 94, 95, 102–104, 107, 109,	\hspace 4298, 454
111, 113, 114, 116, 118–120, 122, 125, 128, 129	\hyperlink 74, 8
enumext . 25, 26, 30, 31, 34, 36–43, 45–56, 58–63, 65, 66,	\hyperlink 2414, 290
68–78, 80, 82, 83, 86–91, 93, 95, 97, 98, 101, 103, 106,	\hypertarget 3
108, 109, 118–120, 122, 125, 126, 128	\hypertarget 41
keyans* 25, 26, 28–32, 36, 39–45, 47, 53, 54, 57, 58, 64, 65,	
67, 68, 70, 78, 83, 88, 92, 102, 104, 105, 113, 125, 127,	I
129	\IfBooleanTF 477
keyanspic 25, 26, 28, 29, 32, 36, 37, 40, 64, 65, 67, 70, 71,	\IfHyperBoolean 39
78, 82–85, 99–102, 127	\IfPackageLoadedTF
keyans 25, 26, 28, 29, 31, 32, 36, 37, 40–43, 45, 47, 49,	\ignorespaces 91
(, ((00 0 0 0	\innutlinana

int commands:	keyanspic 15, <u>3681</u>
\int_add:Nn 3891, 3940	Keys for \anskey provide by enumext:
\int_case:nn 1083, 1227, 1929, 1955, 1994, 2018	break-col
\int_compare:nNnTF 369, 624, 641, 661, 668, 1152,	item-join
1271, 1416, 1432, 2042, 2048, 2513, 2517, 2521, 2529,	item-pos* 74, 76, 79, 81
2575, 2579, 2583, 2780, 2801, 2840, 2845, 2850, 2875,	item-star
2963, 3360, 3371, 3393, 3406, 3445, 3461, 3475, 3489,	item-sym* 74, 76, 79, 81
3553, 3557, 3586, 3611, 3624, 3646, 3650, 3706, 3861,	Keys for anskey* provide by enumext:
3871, 3887, 3910, 3920, 3936, 4093, 4097, 4136, 4146,	break-col
4293, 4302, 4340, 4352, 4535, 4545, 4726, 4853	item-join
\int_compare_p:nNn 237, 247, 259, 260, 274, 275,	item-pos*
1422, 1423, 1935, 1961, 2297, 2307, 2319, 2320, 2335,	item-star
2376, 2552, 2553, 2564, 2565, 2717, 3403	item-sym*
\int_decr:N 3890, 3939	Keys for environments provide by enumext:
\int_eval:n 354, 762, 2196, 2347, 2446, 2858, 2969,	above* 27, 56, 57, 95, 108
2982, 3258, 3302, 3879, 3928, 4457	above 27, 56, 57, 95, 108, 114
\int_from_alph:n	after 45, 46, 97, 109, 114
\int_from_roman:n 728, 742 \int_gadd:Nn 3892, 3941	align 27, 37, 87, 90, 111, 124
	base-fix
\int_gdecr:N 1938, 1943, 1947, 1951, 1964	before* 45, 46, 95, 108, 114
\int_gincr:N 1771, 1776, 2359, 2913, 3002, 3036, 3179,	before
3435, 3577, 3668, 4156, 4234, 4380, 4446 \int_gset:Nn 1987, 3779	below* 27, 56–58, 97, 109
\int_gset.NN 1670, 1677, 1683, 1689, 1697, 1704,	below 27, 56–58, 97, 109, 114
1710, 1716, 3776	check-ans 29–31, 63–67, 70, 82, 84, 97, 109, 113, 126
\int_gzero:N . 325, 326, 327, 1453, 1475, 2054, 2773,	columns-sep
3494, 3629, 4311, 4556	columns 27, 47, 50, 56, 96
\int_if_exist:NTF 1645, 1681, 1687, 1708, 1714, 1892	first
\int_incr:\N\ 2528, 3359, 3548, 3705, 4092, 4155, 4339,	font 37, 87, 90, 111
4379	item-pos*
\int_mod:nn 4304, 4547	itemindent
\int_new:N . 28, 29, 30, 31, 32, 33, 61, 62, 85, 101, 120,	itemsep
140, 141, 146, 147, 148, 150, 161, 167, 168, 169, 170,	labelsep
171, 1647, 1895	labelwidth
\int_set:Nn 722, 726, 728, 1784, 1791, 1803, 1812, 2665,	label 26, 27, 36, 38, 41, 103
3750, 3751, 3806, 3837, 3860, 3866, 3882, 3909, 3915,	lisparindent 92
3931, 4287, 4529, 4722, 4855	list-indent
\int_set_eq:NN 1772, 1777, 3889, 3938	list-offset
\int_sign:n 1989	listparindent 43, 111
\int_step_function:nnN 2313, 2326, 2340	mark-ans
\int_step_function:nnnN 4858	mark-pos
\int_step_inline:nn4774	mark-ref
\int_step_inline:nnn 3752	mini-env 27, 34, 47, 55, 56, 70, 95, 106–109, 114
\int_to_roman:n 206, 2293, 2330	mini-right* 27, 30, 47, 70, 106-109
\int_use:N 347, 352, 353, 1153, 1786, 1793, 1805, 1814,	mini-right 27, 30, 47, 54, 70, 106–109
3258, 3277, 3302, 3379, 3446, 3455, 3470, 3476, 3864,	mini-sep 27, 47, 70, 95
3865, 3877, 3913, 3914, 3926, 5189, 5193, 5199, 5203	no-store
\int_zero:N 4296, 4539	noitemsep
\item . 86, 89, 109, 111, 114, 116, 360, 2214, 2220, 2245, 2251,	nosep 42, 51
2373, 2877, 2880, 3054, 3183, 3737, 4077, 4079, 4326,	parindent
4328, 4444	parsep
\item* 5, 15, 67, 3181	partopsep 42
item-pos* <u>3079</u>	ref
item-sym* <u>3079</u>	resume* 26, 58, 59, 62-64, 70, 97, 109, 120
\itemindent 91	resume 26, 33, 58-64, 70, 97, 109, 120
\itemindent	rightmargin
itemindent	save-ans 28, 33, 59-63, 65, 66, 69-71, 76-78, 82, 84, 89,
\itemsep 100, 101	98, 100, 114, 116, 118, 120, 125
\itemsep	save-key
\itemwidth . 436, 2064, 3340, 3349, 3527, 3536, 3900, 3904,	save-pos
3949, 3953	save-ref 29, 35, 68, 70, 72, 74, 83, 84, 90, 116
K	save-sep
keyans	show-ans
keyans*	show-length

show-pos	mark-ans <u>2057</u>
start* 27, 41, 59	mark-pos
start 27, 30, 41, 59	mark-ref 2057
store-key 69	mini-env 1038
topsep 42	mini-sep 1038
widest	\minipage 364
wrap-ans	\miniright 10, 54, 1414, 1457, 1478, 3492, 3627
wrap-label* 27, 37, 86, 87, 90, 110, 111, 115	mode commands:
wrap-label 27, 37, 86, 87, 90, 110, 111, 115	\mode_if_math:TF 2537, 2591
wrap-opt	\mode_if_vertical:TF 1108, 1136, 1252, 1331
\keys_define:nn 488, 510, 542, 558, 605, 676, 750, 776,	\mode_leave_vertical: 834, 845, 907, 921, 933, 945,
820, 856, 879, 952, 961, 1040, 1057, 1485, 1596, 1839,	2270, 3046, 4252
1900, 2059, 2101, 2137, 2142, 2456, 2607, 2643, 3081,	msg commands:
3097, 3117, 3137, 4585, 4684, 4800, 4808	\msg_error:nn 1459, 1480, 2498, 2531, 2535, 2589,
\keys_if_exist_p:nn 4796, 4797	2697, 3555, 3559, 3648, 3708, 3739, 4095, 4342, 4354,
\l_keys_key_str 76, 79, 2474, 2629, 3105, 3123, 3145,	4711, 4770
4841, 4945	\msg_error:nnn 581, 628, 645, 698, 1418, 1425, 1430,
\keys_precompile:nnN 119, 190, 190, 4587, 4591,	1455, 1477, 1745, 1749, 1864, 2480, 2539, 2557, 2569,
4595, 4599, 4603, 4607, 4827	2577, 2581, 2585, 2593, 2635, 3111, 3129, 3151, 4099,
\keys_set:nn . 502, 836, 847, 1063, 1490, 1495, 1733,	4347, 4575, 4580, 4653, 4758, 4789, 4798, 4836 \msg_error:nnnn 2483, 2511, 2515, 2519, 2523, 2638,
1738, 1825, 1833, 2494, 3373, 3378, 3564, 4110, 4361,	3114, 3132, 3154, 3546, 3644, 3652, 4634, 4837
4639, 4646, 4688, 4693, 4694, 4695, 4696, 4699, 4704,	\msg_error:nnnnn 529, 549, 2080
4705, 4706, 4707, 4708, 4709, 4710, 4742, 4850	\msg_fatal:nn 3361
\keys_set_known:nn 2761	\msg_fatal:nnn 440
keyval commands:	\msg_info:nnn 13, 16, 21, 24, 389, 403
\keyval_parse:NNn 1610, 2167, 4660	\msg_line_context: 4910, 4915, 4920, 4949, 4954,
L	4959, 4974, 4989, 4993, 4997, 5001, 5005, 5009, 5016,
label	5023, 5029, 5043, 5047, 5052, 5056, 5060, 5064, 5069,
Labels provide by enumext:	5073, 5077, 5081, 5086, 5121, 5125, 5130, 5135, 5139,
\Alph* 36	5144, 5220, 5224, 5229, 5234, 5239, 5243, 5247, 5251,
\Roman* 36	5255, 5259, 5263, 5267, 5271
\alph* 36	\msg_log:nnn 1884, 1889, 1894
\arabic* 30, 36	\msg_log:nnnnn 351, 2027, 2032, 2037
\roman* 36	\msg_log:nnnnnn 343
\labelsep 101	\msg_new:nnn 4877, 4881, 4885, 4889, 4894, 4907, 4912,
\labelsep 3722, 3725	4917, 4922, 4931, 4939, 4943, 4947, 4952, 4957, 4972,
labelsep	4987, 4991, 4995, 4999, 5003, 5007, 5011, 5020, 5026, 5032, 5036, 5040, 5045, 5050, 5054, 5058, 5062, 5067,
\labelwidth 36, 101	5032, 5030, 5040, 5045, 5050, 5054, 5050, 5002, 5007, 5071, 5075, 5079, 5084, 5119, 5123, 5128, 5133, 5137,
\labelwidth 3722, 3723 labelwidth 486	5142, 5218, 5222, 5227, 5232, 5237, 5241, 5245, 5249,
\leftmargin	5253, 5257, 5261, 5265, 5269
\leftmargin 90, 3722	\msg_new:nnnn 4898, 5089, 5098, 5107, 5113, 5146,
legacy commands:	5156, 5166, 5176, 5186, 5196, 5206, 5212
\legacy_if:nTF 4222, 4225, 4470, 4473	\msg_term:nnnn . 1848, 1853, 3267, 3277, 3308, 3313
\legacy_if_gset_false:n 374	\msg_term:nnnnn 2008
\legacy_if_set_false:n 4224, 4472	\msg_warning:nn 3491, 3626
\legacy_if_set_true:n 4184, 4209, 4216, 4229, 4408,	\msg_warning:nnnn 2045, 2051, 3215, 3220, 3863, 3876,
4436, 4477	3912, 3925
\linewidth 95	\msg_warning:nnnnn 2003, 2013
\linewidth 3342, 3430, 3529, 3574, 3749, 3809, 3840, 3962,	\multicolsep 96
4017	\multicolsep 3460, 3599
\list 358	NT.
list-indent	N \NaadsTayFarmat
list-offset	\NeedsTeXFormat
\listparindent	\NewDocumentCommand 1414, 2486, 3640, 4559, 4612, 4718,
listparindent	4767, 4843
\lrbox 4240, 4483	\NewDocumentEnvironment . 3318, 3506, 3681, 4065, 4315
M	\newenvsc 2600
\makebox	\newlabel 35
\makebox 2272, 2274, 3048, 4254, 4262, 4266, 4496, 4500	\newlabel 425
\makelabel 86, 87, 90, 102	no-store 1898
\makelabel 86, 89, 3065, 3199	\noindent . 3437, 3579, 3971, 4026, 4078, 4295, 4327, 4538
\makesavenoteenv 407	\nointerlineskip 3437, 3579, 3971, 4026

$\verb"noitemsep" \dots \dots$	resume
\nopagebreak 1119, 1147, 1263, 1342, 1405, 1411	resume*
\normalfont 2445, 2968, 2981, 4456	rightmargin <u>854</u>
nosep	\Roman
P	\Roman
Packages:	\roman
caption 106	\roman 463, 574, 4602
enumext	c.
enumitem 35, 36	\$
expl3 102	\\$
footnotehyper 34	save-ans
hyperref 29, 30, 34, 35, 74, 84, 111, 123	save-key
lua-visual-debug 50	save-ref <u>2057</u>
multicol 25, 123	save-sep
scontents	scan commands:
shortlst 102	\scan_stop: 102, 3735, 4077, 4326, 4570, 4573
\par 1119, 1147, 1263, 1342, 1405, 1411, 1448, 1470, 2422,	scontents internal commands:
3481, 3496, 3616, 3631, 3761, 3989, 4003, 4044, 4058,	\lscontents_fname_out_tl 2653 \scontents_parse_environment_keys:n . 2659
4295, 4309, 4538, 4554 \parbox	\scontents_rescan_tokens:n 2666
\parindent	\\\scontents_storing_bool 2651
\parsep	\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\parsep 3299, 3721, 3728, 3733	seq commands:
parsep	\seq_clear:N
\parskip 4273, 4507	\seq_const_from_clist:Nn 4713
\partopsep 101	\seq_count:N 346, 3694, 4724
\partopsep 3300, 3726	\seq_gclear:N 3769, 3770
partopsep 774	\seq_gput_right:Nn 2203, 3782, 3783
peek commands:	\seq_if_empty:NTF 3788, 4627, 4738
\peek_meaning:NTF 4161, 4175, 4192, 4203, 4385, 4399,	\seq_if_exist:NTF 1887, 4625
4416	\seq_if_in:NnTF 4632
\peek_meaning_remove:NTF 4168, 4392	\seq_item:Nn 2691, 3758
\peek_remove_spaces:n 3187	\seq_map_function:NN 4729
\phantomsection 35	\seq_map_inline:Nn 4640, 4647, 4739, 4740
\phantomsection 414	\seq_map_pairwise_function:NNN 3790
prg commands:	\seq_new:N 121, 122, 124, 138, 162, 163, 1890
\prg_do_nothing:	\seq_pop_left:NN
\prg_new_protected_conditional:Npnn 208 \prg_replicate:nn	\seq_put_right:Nn 3654, 4736, 4752, 4867
\prg_return_false:	\seq_set_from_clist:Nn
\prg_return_true: 211	\seq_set_map_e:NNn
\printkeyans	\seq_use:\Nn
prop commands:	series
\prop_const_from_keyval:Nn 4759	
\prop_count:N 345, 2196, 2347, 2448, 2858, 2971, 2984,	\setcounter 736, 740, 742, 3258, 3302, 3699 \setenumext 6, 120, 4718
4459, 4856	\setenumextmeta
\prop_get:NnNTF 4785	show-ans
\prop_gput_if_not_in:Nnn 2194	
\prop_if_exist:NTF 1882, 4579	show-length950
\prop_item:Nn 4582, 4872	show-pos
\prop_new:N 1885	skip commands:
\ProvidesExplPackage 4	\skip_add:Nn 1088, 1094, 1100, 1110, 1114, 1138, 1142 1232, 1238, 1244, 1254, 1258, 1280, 1333, 1337, 3721
R	\skip_gset:Nn
\raggedcolumns 3469, 3605	\skip_gzero_new:N
\ref	\skip_horizontal:N 922, 934, 946, 4255, 4269, 4503
ref 556, 603, 676	\skip_horizontal:n 908, 2271, 2279, 3047, 3049
\refstepcounter 4231, 4479	4253, 4512
regex commands:	\skip_if_eq:nnTF 1086, 1092, 1098, 1155, 1189, 1230
\regex_match:nnTF 210, 725, 727, 739, 741, 2693	1236, 1242, 1273, 1278, 1299, 1350, 1372, 1502, 1516
\regex_replace_once:nnN 218	1530, 1541, 1552, 1563, 1574, 1585
\renewcommand 591, 636, 653, 706	\skip_new:N 81, 82, 86, 87, 88, 89, 90, 142, 182
$\verb \RenewDocumentCommand 1457, 1478, 3054, 3065, 3183, 3199, \\$	\skip_set:Nn 1071, 1075, 1124, 1128, 1158, 1162, 1166
3737, 3771	1173, 1177, 1181, 1192, 1197, 1201, 1207, 1212, 1217
\ Deauring Dealeage	10== 10=(10== 100, 1000 1000 1=== (1=== (

1313, 1317, 1321, 1352, 1356, 1374, 1378, 1382, 1388,	\tl_gput_right:Nn 457
1392, 1396, 3715, 3729	\tl_greplace_all:Nnn 478
\skip_set_eq:NN 3256, 3298, 3299, 4272, 4273, 4506,	\tl_gset:Nn 265, 266, 280, 281, 1643, 1656, 1661, 1880
4507	2689, 3023, 4198
\skip_use:N 1073, 1077, 1112, 1116, 1120, 1140, 1144,	\tl_gset_eq:NN 474, 3019, 4249
1156, 1175, 1184, 1190, 1195, 1199, 1210, 1214, 1215,	\tl_if_blank:nTF 2478, 2496, 2633, 3109, 3127, 3149
1220, 1256, 1260, 1286, 1503, 1507, 1510, 1517, 1521,	4247, 4835
1524, 3481	\tl_if_empty:NTF . 579, 598, 626, 643, 663, 670, 696
\skip_vertical:N 375, 378	713, 1668, 1673, 1695, 1700, 1758, 1822, 1830, 1859,
\skip_zero:N 3300, 3460, 3599, 3726, 3727	1919, 2210, 2241, 2386, 2730, 2752, 2782, 2811, 2884,
\skip_zero_new:N 1268, 1269, 1270, 1347, 1369, 1370,	2933, 3044, 4425, 4750
1371	\tl_if_empty:nTF 1723
\c_zero_skip . 375, 378, 1086, 1092, 1098, 1156, 1190,	\tl_if_exist:NTF
1230, 1236, 1242, 1273, 1278, 1299, 1350, 1372, 1503, 1517, 1530, 1541, 1552, 1563, 1574, 1585	\tl_if_novalue:nTF 2492, 2808, 2882, 2918, 2998
\small 4590, 4594, 4598, 4602, 4606, 4610	3017, 3025, 3159, 3368, 3692, 3773, 4107, 4359, 4423
\star 3085	\tl_map_inline:Nn 216, 475 \tl_new:N 42, 43, 44, 47, 52, 53, 56, 57, 63, 65, 66, 68, 69
start	102, 103, 104, 110, 111, 112, 113, 114, 115, 116, 117,
start*	118, 119, 123, 125, 126, 127, 129, 132, 133, 145, 153,
\stepcounter	154, 155, 158, 176
str commands:	\tl_put_left::Ne 2719
\c_backslash_str 2539, 4910, 4915, 4920, 4925, 4927,	\tl_put_left:Nn 2218, 2249, 2371, 2713, 2726, 2732
4929, 4934, 4936, 5034, 5038, 5042, 5052, 5056, 5064,	2742, 2950, 2990, 3992, 4047, 4441, 4444
5065, 5069, 5081, 5082, 5086, 5087, 5108, 5110, 5114,	\tl_put_right:Nn 473, 589, 634, 651, 704, 2222, 2253,
5116, 5144, 5207, 5209, 5213, 5215, 5224, 5225, 5229,	2300, 2310, 2323, 2338, 2344, 2349, 2373, 2378, 2385,
5234, 5235, 5239, 5243, 5247	2388, 2398, 2403, 2406, 2412, 2803, 2806, 2813, 2815,
\c_colon_str 2346, 2857, 4570	2842, 2847, 2852, 2855, 2864, 2877, 2880, 2886, 2891,
\c_left_brace_str 5015, 5022, 5028	2901, 4427, 4428
\c_right_brace_str 5015, 5022, 5028	\tl_remove_all:Nn
\str_case:nn 230, 289	\tl_remove_once:Nn 2288, 2827
\str_case:nnTF . 1617, 1625, 2174, 2182, 4667, 4676	\tl_replace_all:Nnn 477,4784
\str_clear:N 3370, 4109	\tl_reverse:N 2287, 2289, 2826, 2828
\str_count:n 225	\tl_set:Nn . 58, 234, 244, 293, 294, 301, 302, 309, 310
\str_if_empty:NTF 1634, 1675, 1702	442, 516, 520, 525, 526, 578, 623, 695, 905, 919, 931,
\str_if_eq:nnTF 3259, 3304, 4769	943, 1757, 1858, 2122, 2132, 2153, 2161, 2442, 2653,
\str_if_in:nnTF 4566	2920, 2965, 2978, 4430, 4453, 4747, 4783, 4851
\str_new:N 128, 177	\tl_set_eq:NN 483, 584, 587, 631, 633, 648, 650, 701
\str_set:Nn 545, 546, 547, 2077, 2078, 2104, 2105	703, 2286, 2825, 2838, 3171, 3175, 3673, 3675
\string 407	\tl_to_str:n
\strutbox . 1160, 1164, 1168, 1179, 1183, 1194, 1203, 1209,	\tl_trim_spaces:n 473, 4736, 4747, 4753, 4769 \tl_use:N . 479, 482, 600, 665, 672, 715, 976, 980, 984
1219, 1232, 1238, 1244, 1275, 1276, 1277, 1280, 1290,	988, 992, 996, 1000, 1004, 1008, 1012, 1016, 1020,
1294, 1303, 1310, 1315, 1323, 1352, 1353, 1356, 1363,	1024, 1028, 1032, 1036, 2276, 2293, 2301, 2312, 2325,
1376, 1384, 1390, 1398, 3731	2330, 2341, 3006, 3012, 3040, 3067, 3068, 3075, 3162,
Т	3166, 3174, 3201, 3202, 3208, 3325, 3512, 3678, 3999,
TEX and LATEX 2ε commands:	4054, 4259, 4270, 4274, 4493, 4504, 4510, 4515, 4615,
\@auxout 423	4616, 4617, 4618, 4619, 4637, 4732, 4849
\@currenvir	token commands:
\protected@write 423	\token_to_str:N 425
tex commands:	topsep 774
\tex_newlinechar:D 2665	\typeout 393, 396, 406, 407
text commands:	
\text_expand:n	\mathbf{U}
\textasteriskcentered 2074, 2091	\u 219, 2694
\thepage 429	unknown
tl commands:	use commands:
\c_space_tl 2939, 4959, 4974, 4997, 5001, 5188, 5189,	\use:N 226, 3072, 3327
5198, 5199, 5259, 5263	\use:n 1608, 2165, 4568, 4658
\tl_clear:N 515, 521, 2055, 2121, 2131, 2152, 2160,	\use_none:nn 417, 4790
2366, 2685, 2686, 2800, 2874, 4422	\usecounter 3257, 3301
\tl_clear_new:N 472	
\tl_const:Nn 50, 456	V
\tl_gclear:N . 337, 338, 339, 1655, 1660, 2775, 3076,	\value 1671, 1677, 1684, 1690, 1698, 1704, 1711, 1717
4007, 4062, 4256	vbox commands:
\tl gclear new N 1642	\vhox set ton:Nn 2007 4052

$\begin{array}{c} \texttt{Vspace} . 835, 846, 1507, 1510, 1521, 1524, 1534, 1536, 1545,\\ 1547, 1556, 1558, 1567, 1569, 1578, 1580, 1589, 1591,\\ 3689, 3700, 4310, 4555 \end{array}$	wrap-label 480 wrap-label* 480 wrap-opt 2057
W	
$ \text{widest} \ \dots \ \phantom{aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa$	Z
wrap-ans	\z 2692