

enumext

ENUMERATE EXERCISE SHEETS

V1.0 2024-11-01^{*}

©2024 by Pablo González[†]

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

 <https://github.com/pablgonz/enumext>

Abstract

This package provides enumerated list environments compatible with *tagging* PDF for creating “*simple exercise sheets*” along with “*multiple choice questions*”, storing the “*answers*” to these in memory using `multicol` and `scontents` packages.

Contents

1	Introduction	1	6	The storage system	11
1.1	Description and usage	2	6.1	Keys for storage system	11
1.2	The concept of left margin	3	6.1.1	Keys for label and ref	12
1.3	User interface	3	6.1.2	Keys for wrap and display	12
1.3.1	Internal counters	3	6.1.3	Keys for debug and checking	12
1.3.2	Public dimension	3	6.2	The command <code>\anskey</code>	13
1.3.3	Support for <code>multicol</code>	4	6.2.1	Keys for <code>\anskey</code>	13
1.3.4	Support for <code>minipage</code>	4	6.3	The environment <code>anskey*</code>	13
1.3.5	The <code>\label</code> and <code>\ref</code> system	4	6.3.1	Keys for <code>anskey*</code>	14
1.3.6	Support for <code>\footnote</code>	4	6.4	The environment <code>keyans</code>	14
2	The environments provided	5	6.4.1	The <code>\item*</code> in <code>keyans</code>	15
2.1	The environment <code>enumext</code>	5	6.5	The environment <code>keyanspic</code>	15
2.2	The environment <code>enumext*</code>	5	6.5.1	Keys for <code>keyanspic</code>	16
2.3	The command <code>\item*</code>	5	6.5.2	The command <code>\anspic</code>	16
2.3.1	Keys for <code>\item*</code>	6	6.6	Printing stored content	17
2.4	The command <code>\item</code> in <code>enumext*</code>	6	6.6.1	The command <code>\getkeyans</code>	17
3	The command <code>\setenumext</code>	6	6.6.2	The command <code>\foreachkeyans</code>	17
4	The command <code>\setenumextmeta</code>	6	6.6.3	The command <code>\printkeyans</code>	18
5	The <code>keyval</code> system	7	7	Full examples	19
5.1	Keys for <code>label</code> and <code>ref</code>	7	8	Tagged PDF examples	21
5.2	Keys for spaces	8	9	The way of non-enumerated lists	22
5.2.1	Vertical spaces	8	10	References	24
5.2.2	Horizontal spaces	9	11	Change history	24
5.3	Keys for add code	9	12	Index of Documentation	25
5.4	Keys for <code>start</code> , <code>series</code> and <code>resume</code>	10	13	Implementation	27
5.5	Keys for <code>multicols</code>	10	14	Index of Implementation	143
5.6	Keys for <code>minipage</code>	11			
5.6.1	The command <code>\miniright</code>	11			
5.6.2	The key <code>mini-right</code>	11			

Motivation and acknowledgments

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

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

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

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

^{*}This file describes a documentation for v1.0, last revised 2024-11-01.

[†]E-mail: pablgonz@educarchile.cl.

License and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License (lpp), 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 T_EX distribution such as T_EX Live or MiK_TE_X. It has been tested with the standard classes provided by L^AT_EX: book, report, article and letter on 10pt, 11pt and 12pt.

The minimum requirement is L^AT_EX release 2024-11-01.

1 Introduction

In the L^AT_EX 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

4. Related to Linux
- (a) You use linux?

(b) Usually uses the package manager?

(c) Rate the following package and class

i. xsim-exam

ii. xsim

iii. exsheets
- (a) $\alpha > \delta$

(b) L^AT_EX2e is cool?

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

1. Factor $x^2 - 2x + 1$

2. Factor $3x + 3y + 3z$

3. True False

4. Related to Linux
- (a) You use linux?

(b) Usually uses the package manager?

(c) Rate the following package and class

i. xsim-exam

ii. xsim

iii. exsheets
- * $(x - 1)^2$

* $3(x + y + z)$

(a) $\alpha > \delta$

(b) L^AT_EX2e is cool?
- * Yes

* Yes, dnf

* 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

4. (a) Yes
- (b) Yes, dnf

(c) i. doesn’t exist for now :(

ii. very good

iii. obsolete
- ⌘

⌘

⌘

⌘

⌘

⌘

⌘

⌘

Another very common thing to use in my work is “multiple choice questions”, for example:

1. First type of questions

2. Second type of questions

3. Third type of questions
- A) value

B) correct

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

D) I and III only

E) I, II, and III

D) value

E) value
4. Question with image and label below:

5. Question with image on right side:
- A) value

B) value

C) value

D) correct

E) value
- 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) $x = 5$

2. D)

3. C) some note
- ⌘ 4. E) A duck

⌘ 5. D) “other note”

⌘

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

- 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.

1.1 Description and usage

The `enumext` package defines enumerated environments using the `list` environment provided by \TeX , 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 $\text{MiK}\text{\TeX}$, use the package manager to install. For manual installation, download `enumext.zip` and unzip it, run `luatex enumext.ins` and move all files to appropriate locations, then run `mktexlsr`. To produce the documentation run `arara enumext.dtx`.

<code>enumext.sty</code>	»	<code>TDS:tex/latex/enumext/</code>
<code>enumext.pdf</code>	»	<code>TDS:doc/latex/enumext/</code>
<code>README.md</code>	»	<code>TDS:doc/latex/enumext/</code>
<code>enumext.dtx</code>	»	<code>TDS:source/latex/enumext/</code>
<code>enumext.ins</code>	»	<code>TDS:source/latex/enumext/</code>

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 `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`.

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` and `\foreachkeyans` to print all *stored content*, `\miniright` for `minipage`, `\setenumext` and `\setenumextmeta` to config [*key* = *val*] options.

1.3.1 Internal counters

The package `enumext` uses internally the `enumXi`, `enumXii`, `enumXiii`, `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 `enumitem`[6] package to customize the references which is activated by the `ref` key (§5.1), the standard \LaTeX `\label` and `\ref` commands work as usual. It also provides an “internal reference” system for the “stored content” by means of the key `save-ref` (§6.1.1) when the key `save-ans` (§6.1) is active.

1.3.6 Support for \footnote

The `enumext*` and `keyans*` environments and the `mini-env` key use the `minipage` environment in their implementation but in a transparent way for the user, i.e. it is only used for typesetting and not directly. The `enumext` package provides an *internal implementation* for the command `\footnote` compatible with the `hyperref` package to work in the same way as if it were used anywhere in the document.

Unfortunately, if *tagging* PDF is not enabled, it will not produce the expected “links” because the internal implementation uses `\footnotetext[⟨number⟩]` and `\footnotemark[⟨number⟩]{⟨text⟩}` and support for these is limited by the `hyperref` package.

The best way to solve this if *tagged* PDF is NOT active 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). Load it is as follows:

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

At the moment the `footnotehyper` package is not compatible with *tagged* PDF.

2 The environments provided

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

enumext	<code>\begin{enumext}[⟨keyval list⟩]</code>	<code>\begin{enumext*}[⟨keyval list⟩]</code>
enumext*	<code>\item ⟨item content⟩</code>	<code>\item ⟨item content⟩</code>
	<code>\item [⟨custom⟩] ⟨item content⟩</code>	<code>\item [⟨custom⟩] ⟨item content⟩</code>
	<code>\item* [⟨symbol⟩] [⟨offset⟩] ⟨item content⟩</code>	<code>\item* [⟨symbol⟩] [⟨offset⟩] ⟨item content⟩</code>
	<code>\end{enumext}</code>	<code>\end{enumext*}</code>

2.1 The environment enumext

The `enumext` is an environment that works in the same way as the standard `enumerate` environment provided by \TeX , `\item` and `\item[⟨custom⟩]` 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 `[⟨key = val⟩]` 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 `shortenumerate` or `tasks` environments provided by the `shortlst`[15] and `tasks`[16] packages, `\item` and `\item[⟨custom⟩]` work as usual. The options can be configured globally using `\setenumext` command and locally using `[⟨key = val⟩]` 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 content” in the environment is placed within a `minipage` environment whose *width* is stored in the dimension `\itemwidth` that NOT includes `labelwidth`, `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 (see §1.3.6 for full support).
- You cannot have any standard list environments like `itemize`, `enumerate`, `description`, `quote`, `quotation`, `verse`, `center`, `flushleft`, `flushright`, `verbatim`, `tabbing`, `trivlist`, `list` and all environments created with `\newtheorem`.

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.
- * 4. This text is in the first level.

2.3 The command \item*

```
\item* \item* [⟨symbol⟩] [⟨offset⟩]
```

The `\item*`, `\item*[⟨symbol⟩]` and `\item*[⟨symbol⟩][⟨offset⟩]` works like the numbered `\item`, but placing a `⟨symbol⟩` to the “left” of the `⟨label⟩` separated from it by the `⟨offset⟩` set by the the *second optional argument*. 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).

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*` = {<symbol>} default: \textasteriskcentered
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-sym*={\star}`.
`item-pos*` = {<rigid length>} default: by levels
Sets the *offset* between the box containing the current <label> defined by `labelwidth` key and the <symbol> 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*.

2.4 The command \item in enumext*

The `\item` command for the `enumext*` environment provides an “first optional argument” `\item(<columns>)` 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
- * 2. The second
3. The third
4. The fourth
- * 5. The fifth item is way too long for this and needs three columns
6. The sixth
7. The seventh
- X 8. The eighth item is way too long for this and needs two columns (196.17749pt)
9. The ninth
- Z 10. The tenth (89.28171pt)

3 The command \setenumext

<code>\setenumext</code>	<code>\setenumext{<key = val>}</code>	<code>\setenumext[<keyans*>]{<key = val>}</code>
	<code>\setenumext[<enumext, level>]{<key = val>}</code>	<code>\setenumext[<print, level>]{<key = val>}</code>
	<code>\setenumext[<enumext*>]{<key = val>}</code>	<code>\setenumext[<print, *>]{<key = val>}</code>
	<code>\setenumext[<keyans>]{<key = val>}</code>	<code>\setenumext[<print*>]{<key = val>}</code>

The command `\setenumext` sets the <keys> 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 <keys> set in the *optional argument* 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 “storage 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.

4 The command \setenumextmeta

<code>\setenumextmeta</code>	<code>\setenumextmeta {<key name>}{<key-one = val, key-two = val, ...>}</code>
	<code>\setenumextmeta*{<key name>}{<key-one = val, key-two = val, ...>}</code>
	<code>\setenumextmeta [<enumext*>]{<key name>}{<key-one = val, key-two = val, ...>}</code>
	<code>\setenumextmeta [<enumext, level>]{<key name>}{<key-one = val, key-two = val, ...>}</code>

The command `\setenumextmeta` adds a new “meta-key” for the environments `enumext` and `enumext*`, the {<key name>} 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 argument* ‘*’ will create the new “meta-key” for the environment `enumext*` and for all levels of the environment `enumext`. For example: `\setenumextmeta*{midsep}{topsep=3pt, partopsep=0pt}` will create a new key `midsep` available for all levels of the `enumext` environment and the `enumext*` environment and we can use it like any other key so `\begin{enumext}[midsep]` and `\begin{enumext*}[midsep]` will be valid.

5 The keyval system

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 $\langle keys \rangle$ 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.

5.1 Keys for label and ref

`mode-box` $\langle value forbidden \rangle$ default: *not used*

This is a “*switch-key*” that does not receive an argument and is “*only*” available for the “*first level*” of the `enumext` environment and the `enumext*` environment. When this is set the `label`, `font`, `wrap-label` and `wrap-label*` keys are executed within `\makebox` for the `enumext` and `keyans` environments.

- This key is intended for compatibility with *tagged PDF* and is forcibly “*enabled*” when `\DocumentMetadata` is present. If you want to get the same document output whether `\DocumentMetadata` is active or not, you must enable this key.

- In the `enumext*` and `keyans*` environments `\makeLabel` are redefined using `\makebox` by default. If `enumext` or `keyans` is used in the `enumext*` environment the key must be activated manually.

`label` = { $\langle \backslash alph* | \backslash Alph* | \backslash arabic* | \backslash roman* | \backslash Roman* \rangle$ } 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 $\langle \backslash alph* \rangle$, for third level are `\roman*`, and for fourth level are `\Alph*`. For `keyans` and `keyans*` environments the default value is `\Alph*`.

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

`labelsep` = { $\langle rigid length \rangle$ } default: `0.3333em`

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

`labelwidth` = { $\langle rigid length \rangle$ } default: *by label*

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

`widest` = { $\langle integer | string \rangle$ } default: *empty*

Sets the `labelwidth` key pass the $\langle integer \rangle$ or converting the $\langle string \rangle$ of the form `\Alph`, `\alph`, `\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 $\langle label \rangle$ defined by `label` key. For example `font={\bfseries\small}`.

`align` = { $\langle left | right | center \rangle$ } default: *left*

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

`wrap-label` = { $\langle code \{ \#1 \} \text{ more code} \rangle$ } default: *empty*

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

```
\NewDocumentCommand \mywrap { s m }
{
  \IfBooleanTF{\#1}
  {
    {\textcolor{red}{\textbf{Q}}\textcolor{blue}{\textbf{.}}\textcolor{gray}{\#2}}
    {\textcolor{blue}{\textbf{Q}}\textcolor{red}{\textbf{.}}\textcolor{gray}{\#2}}
  }
}
```

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

`wrap-label*` = { $\langle code \{ \#1 \} \text{ more code} \rangle$ } default: *empty*

The same as the `wrap-label` key but also applies on `\item[custom]`.

`ref = {⟨code⟩ {⟨\alph*⟩⟨\Alph*⟩⟨\arabic*⟩⟨\roman*⟩⟨\Roman*⟩ more code⟩}` 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{⟨\alph*⟩}` 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{enumxi}` to indicate the count of the first level instead of using `\theenumxi`.

5.2 Keys for spaces

`show-length = {⟨true⟩⟨false⟩}` 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 = {⟨rubber length⟩⟨rigid length⟩}` 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 = {⟨rubber length⟩⟨rigid length⟩}` 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`.

- In the `enumext*` and `keyans*` environments this value is passed to `\parskip` within the `minipage` environment where “item content” is placed.

`partopsep = {⟨rubber length⟩⟨rigid length⟩}` 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. T_EX will enter *⟨vertical mode⟩* and apply this value to the “top” and “bottom” the environment or nested level.

`itemsep = {⟨rubber length⟩⟨rigid length⟩}` 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`.

- In the `enumext*` and `keyans*` environments this value corresponds to the separation between rows.

`noitemsep` *⟨value forbidden⟩* default: *not used*

This is a “meta-key” that does not receive an argument. Set `itemsep` and `parsep` equal to `0pt` 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 `0pt` the entire level of environment.

`base-fix` *⟨value forbidden⟩* default: *not used*

This is a “switch-key” that does not receive an argument available *only* for the “first level” of environment `enumext`. Fix the *baseline* when an environment `enumext` is nested in `enumext*` and there is no material between the `\item` and the start of the environment for example `\item \begin{enumext}` within the environment `enumext*`. Internally sets the keys `topsep`, `above` and `above*` at `0pt`.

- This key is provided as a way to work around this minor issue, but you should be aware that if for some reason you have the `itemindent` key set in the `enumext*` environment it will be lost and you will need to adjust it using the `list-offset` key in the `enumext` environment.

- The following $\langle keys \rangle$ should be used with “caution”, they are intended to be used at the “top” and “bottom” of the environment when the `columns` or `mini-env` keys do not provide adequate *vertical spaces*. The values passed can be *rubber* or *rigid* lengths, the way they are applied is the way you differ, using the star ‘*’ $\langle keys \rangle$ applies `\vspace*` so that \LaTeX 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 “above” the environment without hindering the value of the `topsep` key. The space is added with `\vspace` so is “discordable”.

`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 “above” the environment without hindering the value of the `topsep` key. The space is added with `\vspace*` so is “not discordable”.

`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 “discordable”.

`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 discordable”.

5.2.2 Horizontal spaces

`list-offset = { $\langle rigid length \rangle$ }` default: `0pt`

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

`list-indent = { $\langle rigid length \rangle$ }` default: `labelwidth + labelsep`

Sets the *indentation* of the whole environment under the box defined by `labelwidth` and `labelsep` keys. Internally sets the value of `\leftmargin` and `\itemindent` for the current level. If `list-indent=0pt` is set in the environments `enumext` and `keyans` 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”.

- The `enumext*` and `keyans*` environments are implemented using `\makebox` and `minipage` which causes “list indent” to always be equal to the value passed to `labelwidth` plus `labelsep`. Passing a value to this key is equivalent to setting the value for the `list-offset` key.

`itemindent = { $\langle rigid length \rangle$ }` default: `0pt`

Sets the extra *horizontal indentation*, beyond `labelsep`, of the “first line” off each `\item` that is not followed by a “blank line” or the `\par` command. This value must be greater than or equal to `0pt` and is applied internally using `\hspace` without modifying the value of `\itemindent`.

- This key is intended for the `enumext*` and `keyans*` environments where, by their implementation, it is not possible to adjust `labelwidth` and `list-indent` without modifying the output. If you use `enumext` or `keyans` and want to get around the *blank line* limitation or the `\par` command followed by `\item` you can modify `labelwidth` and `list-indent` and get the same effect.

`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 `0pt`. Internally sets the value of `\rightmargin` for the current level.

`listparindent = { $\langle rigid length \rangle$ }` default: `0pt`

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

- In the `enumext*` and `keyans*` environments this value is passed to `\parindent` within the `minipage` environment where “item content” is placed.

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 \LaTeX which is defined (simplified) as plain form `\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}{ $\langle arg one \rangle$ }{ $\langle arg two \rangle$ }{ $\langle code \rangle$ }`.

`before*` = {`<code>`} default: *not used*
 Execute {`<code>`} “before” the environment starts. The {`<code>`} must be passed between braces, is executed “before” performing all calculations related to the `list parameters` and [`<key = val>`] sets in the environment that is, before the arguments defining the environment are executed: {`<code>`}\begin{list}{`<arg one>`}{`<arg two>`}.

`first` = {`<code>`} default: *not used*
 Executes {`<code>`} when “starting” the environment. The {`<code>`} 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{list}{`<arg one>`}{`<arg two>`}{`<code>`}\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.
- 🟢 In the `enumext*` and `keyans*` environments this key is executed after the `listparindent`, `parsep` and `itemindent` keys within the `minipage` environment in which the “item content” is placed.

`after` = {`<code>`} default: *not used*
 Execute {`<code>`} “after” finishing the environment. The {`<code>`} must be passed between braces.

5.4 Keys for start, series and resume

`start` = {`<integer | integer expression>`} default: `1`
 Sets the *start value* of the numbering on the current level. The {`<integer expression>`} 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}}`.

`start*` = {`<integer | string>`} default: *not used*
 Sets the *start value* of the numbering on the current level. Internally `<string>` 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`.

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` = {`<series name>`} default: *not used*
 Stores the *keys* of the *optional argument* of the “first level” of the environment in which it is executed in {`<series name>`} which is used as an argument in the key `resume`. The `<keys>` stored in {`<series name>`} are not cumulative and are overwritten if the same {`<series name>`} is used again.

`resume` = {`<series name>`} default: *not used*
 Sets the *start value* and *options* for the “first level” continuing the numbering of the environment in which the `series={<series name>}` key was executed. If passed *without value* this will only set *start value* continue the numbering from the last environment in which `series={<series name>}` or `resume={<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*` `<value forbidden>` default: *not used*
 Sets the *start value* and *options* for the “first level” continuing the numbering of the environment in which the `series={<series name>}` or `resume={<series name>}` 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.

- 🟢 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={<series name>}` it will have hierarchy in the `<keys>` that are saved in {`<series name>`}, in order to establish the value of a `<key>` already saved in {`<series name>`} it must be placed to the “right” of `resume={<series name>}`, 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*.

5.5 Keys for multicols

`columns` = {`<integer>`} 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` = {`<rigid length>`} default: *by level*
 Set the *space between columns* used by the `multicols` environment within the environment. Internally sets the value of `\columnsep`, by default its value is equal to the sum of the values set in the keys `labelwidth` and `labelsep` of the current level.

5.6 Keys for minipage

`mini-env = {⟨rigid length⟩}`

default: *not used*

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

`mini-sep = {⟨rigid length⟩}`

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={⟨rigid length⟩}] ⟨item's before⟩ \item \miniright ⟨content⟩ \end{enumext}
\begin{enumext}[mini-env={⟨rigid length⟩}] ⟨item's before⟩ \item \miniright*⟨content⟩ \end{enumext}
```

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 L^AT_EX justification is maintained in the `minipage` on the “right side”.

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 = {⟨content⟩}`

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 `{⟨content⟩}` must be passed between braces.

`mini-right* = {⟨content⟩}`

default: *not used*

Same as above, but *without* starting with `\centering`.

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 `⟨key⟩` is “*active*” the `\anskey` command and the environments `anskey*`, `keyans`, `keyans*` and `keyanspic` are available.

<pre>\begin{enumext}[save-ans={⟨store name⟩}] \item Text \anskey{answer} \item Text \begin{keyans} ... \end{keyans} \end{enumext}</pre>	<pre>\begin{enumext}[save-ans={⟨store name⟩}] \item Text \anskey{answer} \item Text \begin{keyanspic} ... \end{keyanspic} \end{enumext}</pre>
---	---

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

For security reasons the *optional argument* of the inner levels or the nested environment are *filtered* by excluding all `⟨keys⟩` related to the “*storage system*” (§6.1) along with the keys `mini-env`, `mini-sep`, `mini-right`, `mini-right*`, `series`, `resume` and `resume*` when storing in *sequence* `{⟨store name⟩}` set by `save-ans` key.

6.1 Keys for storage system

The only `⟨keys⟩` available for all levels of the `enumext` environment and the `enumext*` environment are `no-store` and `save-key`, the rest of the `⟨keys⟩` 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 = {⟨store name⟩}`

default: *not set*

Sets the *name* of the *sequence* and *prop list* in which the `{⟨contents⟩}` will be “*stored*” by `\anskey` and `anskey*` in `enumext` and `enumext*` environments and the current `⟨labels⟩` for `\item*` and `\anspic*` in the environments `keyans`, `keyans*` and `keyanspic`. If the *sequence* or *prop list* `{⟨store name⟩}` does not exist, it will be created globally and will not be *overwritten* if the key is used again.

`save-key = {⟨key list⟩}`

default: *not set*

This key *overrides* the default “*stored keys*” of the *optional argument* of the inner levels or nested environment that will be passed to the *sequence*. The `⟨key list⟩` passed to this key ignores any `⟨keys⟩` in the “*stored structure*” and must be passed between braces. For example, if we execute at a second level:

```

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

```

The “*stored keys*” by default in the *sequence* $\{\langle store name \rangle\}$ would be `nosep`, `columns=2`, but using the key `save-key={columns=3}` will overwrite and the “*stored key*” in the *sequence* $\{\langle store name \rangle\}$ are only `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 `\item*` and `\anspic*` in the environments `keyans`, `keyans*` and `keyanspic` and storing them in the *sequence* and *prop list* $\{\langle store name \rangle\}$ set by `save-ans` key. The $\{\langle text symbol \rangle\}$ must always be passed between braces, whitespace ‘`␣`’ 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 true | false \rangle}` default: false

Activates the “*internal label and ref*” mechanism for referencing “*stored content*” in *prop list* $\{\langle store name \rangle\}$ 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 *prop list* $\{\langle store name \rangle\}$ 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` in *prop list* `test` within the environment in which the key `save-ans=test` was set.

`mark-ref = {\langle symbol \rangle}` default: \%

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 | 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: \textasteriskcentered

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 | 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 | 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 | 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 “switch-key” that does not receive an argument and disables the “stored structure” 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*[*<keys>*]{*<content>*}

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 *hyperref*[8] package is detected, *\hyperlink* and *\hypertarget* will be used, otherwise the usual “label and ref” system provided by *LaTeX* will be used.

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 <content>*, the following *<keys>* allow modifying the way in which it is “stored” in the *sequence*.

break-col *<value forbidden>* default: *not used*

Stores $\{\langle content\rangle\}$ in the *sequence* $\{\langle store\ name\rangle\}$ of the form *\columnbreak \item <content>*.

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

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

item-star *<value forbidden>* default: *not used*

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

item-sym* $\{\langle symbol\rangle\}$ default: *not set*

Sets the *symbol* for *\item** when using the key *item-star* and stores $\{\langle content\rangle\}$ in the *sequence* $\{\langle store\ name\rangle\}$ of the form *\item*[\langle symbol\rangle] <content>*. The *symbol* can be in text or math mode, for example *item-sym*={\\$ \ast \\$}* stores *\item*[\\$ \ast \\$] <content>*.

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

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

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{\third answer}
  \item Text containing our instructions or questions. \anskey{\fourth answer}
\end{enumext}
```

- * 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** *\begin{anskey*}[\langle key = val\rangle] \langle body content\rangle \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 *LaTeX* 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.
- ©2024 by Pablo González L
- 13 / 158

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

6.3.1 Keys for anskey*

The `anskey*` environment uses the same `<keys>` 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 = {<file.ext>}` default: not used

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

`overwrite = {<true | false>}` default: false

Sets whether the `<file.ext>` generated by `write-env` from the `anskey*` environment will be rewritten.

`force-eol = {<true | false>}` 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 `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.

[5]

First answer with verbatim
6. Text containing our instructions or questions.

(a) Question.

[6]

second answer
7. Text containing our instructions or questions.

[7]

third answer
8. Text containing our instructions or questions.

[8]

fourth answer

6.4 The environments keyans and keyans*

keyans

\begin{keyans}[<key = val>] \item \item[<custom>] \item* \item*[<content>] \end{keyans}

keyans*

\begin{keyans*}[<key = val>] \item \item[<custom>] \item* \item*[<content>] \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` and `\item[<custom>]` work in the usual and the command `\item(<columns>)` is available for the `keyans*` environment.

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

```
\begin{enumext}[save-ans=test]
  \item <item content>
    \begin{keyans}[<key = val>]
      \item <item content>
      \item [<custom>] <item content>
      \item* <item content>
      \item* [<content>] <item content>
    \end{keyans}
  \end{enumext}
```

```
\begin{enumext}[save-ans=test]
  \item <item content>
    \begin{keyans*}[<key = val>]
      \item <item content>
      \item [<custom>] <item content>
      \item* <item content>
      \item* [<content>] <item content>
    \end{keyans*}
  \end{enumext}
```

The `<keys>` 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[<keyans>]{<key = val>}` or `\setenumext[<keyans*>]{<key = val>}`. If the *optional argument* is not passed or the `<keys>` 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 `<label>` which will be set to `label=\Alph*`.

6.4.1 The `\item*` in `keyans` and `keyans*`

```
\item* \item*
\item* [<content>]
```

The `\item*` and `\item* [<content>]` command “store” the current `<label>` set by `label` key next to the *optional argument* `<content>` in *sequence* and *prop list* `{<store name>}` 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.

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
    \item* [<note>] Correct choice
    \miniright
    \includegraphics[scale=0.25]{example-image-a}
    Some text
  \end{keyans}
\end{enumext}
```

1. Text containing a question.

A) Choice

* B) Correct choice

C) Choice

D) Choice

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`
- ```
keyanspic \begin{keyanspic}[<key = val>] \anspic* [<content>]{<drawing or tabular>} \end{keyanspic}
```
- The `keyanspic` environment is an “enumerated list” environment activated by the `save-ans` key that has the same configuration for “spacing” and `<label>` as the `keyans` environment that uses the `\anspic` command instead of `\item`. It is intended for placing *drawings or tabular* with `<label>` centered *above* or *below* in a *single line* or *upper and lower* layout style.
- When the `keyanspic` environment is used *without keys* the `<labels>` are centered *below* the *drawings or tabular* in a *single line* layout style.
- ©2024 by Pablo González L
- 15 / 158

A representation of the output can be seen in the figure 6.



Figure 6: Representation of the `keyanspic` environment with `layout-sty={⟨3, 2⟩}` in `enumext`.

This environment cannot be nested and must always be at the “first level” of the `enumext` environment, the `\item` command is disabled and keys cannot be set using `\setenumext`.

### 6.5.1 Keys for `keyanspic`

`label-pos = {⟨above | below⟩}` default: *below*

Set the *position* of `⟨label⟩` to be centered “above” or “below” *drawings or tabular* when the `\anspic` command is executed.

`label-sep = {⟨rubber length | rigid length⟩}` default: *internal adjustment*

Set the *vertical spacing* between the `⟨label⟩` centered “above” or “below” and *drawings or tabular* when running the `\anspic` command.

`layout-sty = {⟨n° upper , n° lower⟩}` default: *not set*

Set the *number of drawings or tabular* that will be distributed “upper” and “lower” within the environment when executing the `\anspic` command. The value must be passed in braces and if not set or the `⟨n° lower⟩` is omitted the *drawings or tabular* will be put on a *single line*.

`layout-sep = {⟨rubber length | rigid length⟩}` default: *adjusted parsep from keyans*

Set the *vertical separation* between the number of *drawings or tabular* placed at the “upper” and “lower” within the environment when executing the `\anspic` command. Internally adjusts the `parsep` value taken from the `keyans` environment.

`layout-top = {⟨rubber length | rigid length⟩}` default: *adjusted topsep from keyans*

Set the *vertical space* added to both the top and bottom of the environment. Internally adjust the value of `topsep` taken from `keyans` environment.

### 6.5.2 The command `\anspic`

---

`\anspic` `\anspic{⟨drawing or tabular⟩}`  
`\anspic* [⟨content⟩]{⟨drawing or tabular⟩}`

The `\anspic` command take three arguments, the *starred argument* ‘`*`’ store the current `⟨label⟩` next to the *optional argument* `⟨content⟩` in *sequence* and *prop list* `{⟨store name⟩}` 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 and labels below.

 \begin{keyanspic}[layout-sty={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}


 \item Question with images and labels above.


 \begin{keyanspic}[label-pos=above, layout-sty={3,2}, layout-sep=0.25cm]
 \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}
```


```
\item Question with images and labels below.


\begin{keyanspic}
\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 and labels below.

  
A)


  
B)

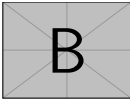
  
C)


  
D)


  
\* E)[note]


2. Question with images and labels above.

A)  



B)  


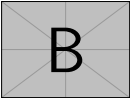
C)  



D)  



\* E)[note]  



3. Question with images and labels below on a single line.

  
A)

  
B)

  
C)

  
D)

  
\* E)[note]

Remember to pass the `alt={description}` key to the `\includegraphics` command when creating a *tagged* PDF.

6.6 Printing stored content

6.6.1 The command `\getkeyans`

```
\getkeyans \getkeyans{<store name> : <position>}
```

The command `\getkeyans` prints the “stored content” in *prop list* `{<store name>}` defined by `save-ans` key in the `<position>` returned by the `show-pos` key. The “stored content” can only be accessed *after* it is stored, if `{<store name>}` does not exist the command will return an error.

The form taken by the argument `{<store name> : <position>}` is the same as that used to generate the “internal label and ref” system when `save-ref` key are active, so to refer to a “stored content”. For example `\getkeyans{test:4}` will return the “stored content” at position `4` of the environment in which the key `save-ans=test` was set.

6.6.2 The command `\foreachkeyans`

```
\foreachkeyans \foreachkeyans[<key = val>]{<store name>}
```

The command `\foreachkeyans` goes through and executes the command `\getkeyans` on the contents in *prop list* `{<store name>}`. If you pass without options run `\getkeyans` on all contents in *prop list* `{<store name>}`.

Options for command

`sep = {<code>}` default: `{;}`

Establishes the *separation* between “each” `{<content>}` stored in *prop list* `{<store name>}`. For example, you can use `sep={\[\ 10pt]}` for vertical separation of stored contents.

`step = {<integer>}` default: `1`

Sets the *step* (increment) applied to the value set by key `start` for “each” `{<content>}` stored in *prop list* `{<store name>}`. The value must be a *positive integer*.

`start = {⟨integer⟩}` default: 1  
 Sets the *position* of the *prop list* {⟨store name⟩} from which execution will start. The value must be a ⟨positive integer⟩.

`stop = {⟨integer⟩}` default: 0  
 Sets the *position* of the *prop list* {⟨store name⟩} from which execution will finish. The value must be a ⟨positive integer⟩.

`before = {⟨code⟩}` default: empty  
 Sets the {⟨code⟩} that will be executed ⟨before⟩ each {⟨content⟩} stored in *prop list* {⟨store name⟩}. The {⟨code⟩} must be passed between braces.

`after = {⟨code⟩}` default: empty  
 Sets the {⟨code⟩} that will be executed ⟨after⟩ each {⟨content⟩} stored in *prop list* {⟨store name⟩}. The {⟨code⟩} must be passed between braces.

`wrapper = {⟨code {#1} more code⟩}` default: empty  
 Wraps the {⟨content⟩} stored in *prop list* {⟨store name⟩} referenced by {#1}. The {⟨code⟩} must be passed between braces. For example `\foreachkeyans[wrapper={\makebox[1em][l]{#1}}]{⟨store name⟩}`.

### 6.6.3 The command `\printkeyans`

---

```
\printkeyans {⟨store name⟩}
\printkeyans[⟨keys⟩]{⟨store name⟩}
\printkeyans*[⟨keys⟩]{⟨store name⟩}
```

---

The command `\printkeyans` prints “all stored content” in *sequence* {⟨store name⟩} defined by `save-ans` key placing this inside the `enumext` or `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 ⟨keys⟩ in the “first level” of the environment in which the “stored content” of the *sequence* {⟨store name⟩} 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* {⟨store name⟩} 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*{⟨store name⟩}` and the *sequence* {⟨store name⟩} already contains any `enumext*` environment an error will be returned as we cannot nest.
- If we execute `\printkeyans*{⟨store name⟩}` and the *sequence* {⟨store name⟩} contains any `enumext` environments, they will start with the ⟨keys⟩ 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{⟨store name⟩}` and the *sequence* {⟨store name⟩} contains any environment `enumext*`, they will start with the ⟨keys⟩ 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[⟨print, 1⟩]{⟨keys⟩}` and `\setenumext[⟨print*⟩]{⟨keys⟩}`.

If we need to set the ⟨keys⟩ for the environment `enumext` “saved” in the *sequence* {⟨store name⟩} we will use `\setenumext[⟨print, level⟩]{⟨keys⟩}` and if we need to set the ⟨keys⟩ for the environment `enumext*` “saved” in the *sequence* {⟨store name⟩} we will use `\setenumext[⟨print, *⟩]{⟨keys⟩}`.

#### Example

```
\begin{enumext}[save-ans=sample,columns=1,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}
 \end{enumext}
\end{enumext}
```



```
\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  $3x + 3y + 3z$ .

[1]

2. True False

(a)

[2]

3. Related to Linux

(a) You use linux?

[3]

(b) Rate the following package and class

i.

[4]

ii.

[5]

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 ✖
- (b) i. very good ✖
- ii. obsolete ✖

## 7 Full examples

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

```
$ pdftdetach -saveall enumext.pdf
```

and then you can use the excellent [arara](#)<sup>1</sup> tool to compile them.

### Example 1

Adapted from the response given by Enrico Gregorio in [Squares for answer choice options and perfect alignment to mathematical answers](#) .

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

☐ A 36 km/h.

☐ B 360 km/h.

☐ C 27,8 km/h.

☐ D  $3,60 \times 10^8$  km/h.

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

☐ A 36 km/h.

☐ B 360 km/h.

☐ C 27,8 km/h.

☐ D  $3,60 \times 10^8$  km/h.

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

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

☐ B  $1 \text{ \AA} = 1 \times 10^{-5} \text{ fm}$ .

☐ C  $1 \text{ \AA} = 1 \times 10^{-15} \text{ fm}$ .

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

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

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

☐ B  $1 \text{ \AA} = 1 \times 10^{-5} \text{ fm}$ .

☐ C  $1 \text{ \AA} = 1 \times 10^{-15} \text{ fm}$ .

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


1. B

2. A

3. B

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 \(cross mark\)](#) .

<sup>1</sup>The cool TeX automation tool: <https://www.ctan.org/pkg/arara>

©2024 by Pablo González L

19 / 158

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  $3,60 \times 10^8$  km/h.
2. In fisica nucleare si usa l'angstrom (simbolo:  $1 \text{ \AA} = 1 \times 10^{-10}$  m) e il fermi o femtometro ( $1 \text{ fm} = 1 \times 10^{-15}$  m). Qual è la relazione tra queste due unità di misura?

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

B  $1 \text{ \AA} = 1 \times 10^{-5} \text{ fm}$ .

C  $1 \text{ \AA} = 1 \times 10^{-15} \text{ fm}$ .

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

A 36 km/h.

✓ B 360 km/h.

C 27,8 km/h.

D  $3,60 \times 10^8$  km/h.
4. In fisica nucleare si usa l'angstrom (simbolo:  $1 \text{ \AA} = 1 \times 10^{-10}$  m) e il fermi o femtometro ( $1 \text{ fm} = 1 \times 10^{-15}$  m). Qual è la relazione tra queste due unità di misura?

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

B  $1 \text{ \AA} = 1 \times 10^{-5} \text{ fm}$ .

C  $1 \text{ \AA} = 1 \times 10^{-15} \text{ fm}$ .

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

✖ 2. A

✖ 3. B

✖ 4. A

Example 3

A “simple multiple choice” test 📄

1. First type of questions

- A) value

C) value
- B) correct

D) value

2. Second type of questions

- I.  $2\alpha + 2\delta = 90^\circ$

II.  $\alpha = \delta$

III.  $\angle EDF = 45^\circ$

A) I only

B) II only

C) I and II only

D) I and III only

E) I, II, and III

3. Third type of questions

- (1)  $2\alpha + 2\delta = 90^\circ$

(2)  $\angle EDF = 45^\circ$

A) value

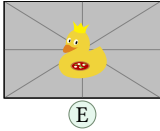
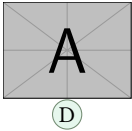
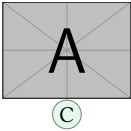
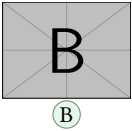
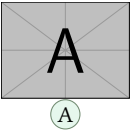
B) value

C) value

D) value

E) value

4. Question with image and label below:



5. Question with image on right side:

- A) value

B) value

C) value

D) correct

E) value



Test keys

1. B),  $x = 5$

✖ 4. E, A duck

2. D

✖ 5. D, other note

3. C, some 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)  $\text{\LaTeX}$ ze is cool?

Related to Linux

- (a) You use linux?

(b) Usually uses the package manager?

- (c) Rate the following package and class
- i. xsim-exam
  - ii. xsim
  - iii. exsheets

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

1.  $(x - 1)^2$

2.  $3(x + y + z)$

3. (a) False

(b) Very True!

4. (a) Yes
- ⌘

(b) Yes, dnf

⌘

(c) i. doesn't exist for now :(

⌘

ii. very good

⌘

iii. obsolete

⌘

Example 5

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

1

Which choice best describes what happens in the passage?

A) One character argues with another character who intrudes on her home.

B) One character receives a surprising request from another character.

C) One character reminisces about choices she has made over the years.

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

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.

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

2

Which choice best describes what happens in the passage?

A) One character argues with another character who intrudes on her home.

B) One character receives a surprising request from another character.

C) One character reminisces about choices she has made over the years.

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

4

Which choice best describes what happens in the passage?

A) One character argues with another character who intrudes on her home.

B) One character receives a surprising request from another character.

C) One character reminisces about choices she has made over the years.

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

1. A)

2. C)

3. B)









4. D)





8 Tagged PDF examples

This section is just to show the compatibility of enumext with tagged PDF using lualatex. The attached files here are just for testing and are intended as examples and, in a way, to simplify the time of Matthew Bertucci (@mbertucci) when he sees this excellent package and adds it to The LaTeX Tagged PDF repository.

To compile the tests with lualatex-dev the packages multicol, scontents, unicode-math, geometry, graphicx, luamml and hyperref are required along with the line:

```
\DocumentMetadata
{
 lang = en-US, pdfversion = 2.0, pdfstandard = ua-2,
 testphase = {phase-III, math, title, table, firstaid},
}
```

- ◆ All examples have been checked using veraPDF together with ngpdf.
- The file enumext-01.tex contains the basic tests for the enumext and enumext\* environments and the nesting between them plus the use of the label, labelwidth, labelsep, ref, align and wrap-label keys. Source file  and tagged PDF .
  - The file enumext-02.tex contains the tests for the enumext and enumext\* environments and the support for minipage and multicol environments using the keys columns, columns-sep, mini-env, mini-right and \miniright command. Source file  and tagged PDF .
  - The file enumext-03.tex contains the tests for the enumext and keyanspic environments activated by the save-ans key together with the save-sep and save-ref keys and the \printkeyans command. Source file  and tagged PDF .
  - The file enumext-04.tex contains the tests for the \anskey command and the anskey\* environment activated by the save-ans key along with the \getkeyans and \printkeyans commands. Source file  and tagged PDF .

- The file `enumext-05.tex` contains the tests for the environments `keyans`, `keyans*` and `keyanspic` activated by the key `save-ans` together with the keys `no-store` and `show-ans` and the commands `\setenumext`, `\setenumextmeta`, `\printkeyans` and `\foreachkeyans`. Source file  and tagged PDF .
- The file `enumext-06.tex` contains the tests for the environments `enumext` and `enumext*` for *fake itemize* and *description*. Source file  and tagged PDF .

9 The way of non-enumerated lists

It is possible to use (or abuse) the `enumext` and `enumext*` environments to mimic *non-enumerated* list environments such as `itemize` and `description`, clearly the  $\langle keys \rangle$  to “store answers”, the `keyans`, `keyans*` and `keyanspic` environments lose their sense and it is not the focus of `enumext` 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={\some}` 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  $\TeX$  for the four levels `\textbullet`, `\textendash`, `\textasteriskcentered` and `\textperiodcentered` together with the `nosep` key to reduce the vertical spaces in the left side example and set the `label` key in *mathematical mode* for the right side as `\ast`, `\diamond`, `\circ` and `\star` for the four levels together with the `nosep` key

- First level item
    - Second level item
      - \* Third level item
        - Fourth level item
  - First level item
- \* First level item
    - ◊ Second level item
      - Third level item
        - ★ Fourth level item
  - \* First level item

Fake description environment

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

- Something** A short one-line description.

This is an entry *without* a label.

**Something** A short *one-line* description text.

**Something long** A much *longer* description text may take more than one line or more than one paragraph.

    Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

If we add `list-indent=0pt` 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.
- 🔗 When *tagged PDF* is active the default `description` style is NOT available due to the redefinition of `\makelabel` for the `align` key which uses `\makebox` in this case, meaning that `\item[\langle content \rangle]` will not extend beyond `\labelwidth` which causes overlaps,

Description indented by label

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

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

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

- Something** A short one-line description.

This is an entry *without* a label.

**Something** A short one-line description.

**Something long** A much longer description. Lorem ipsum dolor sit amet, consectetur 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, consectetur 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, consectetur 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` and `wrap-label*` keys 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 }
{%
 \SuspendTagging{\parbox}%
 \IfBooleanTF{#1}
 {%\strut\smash{\parbox[t]{\labelwidth}{\raggedright{#2}}}}%
 {%\strut\smash{\parbox[t]{\labelwidth}{\raggedleft{#2}}}}%
 \ResumeTagging{\parbox}%
}
```

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

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

**LoNg** 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 `l3prop[11]` and `l3seq[11]` modules together with the `hyperref[8]` and `enumitem[6]` 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 typesetting a document with  $\LaTeX$ , 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.



## Why has it taken so long?

One of the setbacks, beyond my laziness, was including compatibility with *tagged* PDF. To be honest, it's something I never considered at any point, but I firmly believe that being able to create *accessible documents* provides a great opportunity in the world of mathematics education. From my perspective as a *high school* teacher, beyond theorems and deep mathematics, the use of exercise lists is one of the most common things. Being able to open the way to work in parallel with those who have different abilities is really important and I regret not having looked into this in the past. I hope that `enumext` serves this purpose and inspires more users and authors to follow this path.

## 10 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] MITTELBAACH, FRANK. “An environment for multicolumn output”. Available from CTAN, <https://www.ctan.org/pkg/multicol>, 2024.
- [4] GONZÁLEZ, PABLO. “scontents - Stores  $\LaTeX$  contents in memory or files”. Available from CTAN, <https://www.ctan.org/pkg/scontents>, 2024.
- [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. “ $\LaTeX 2_{\epsilon}$ : An Unofficial Reference Manual”. Available from CTAN, <https://ctan.org/pkg/latex2e-help-texinfo>, 2024.
- [8] The  $\LaTeX$  Project. “Extensive support for hypertext in  $\LaTeX$ ”. 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  $\LaTeX$  Project. “The expl3 package”. Available from CTAN, <https://www.ctan.org/pkg/l3kernel>, 2024.
- [11] The  $\LaTeX$  Project. “The  $\LaTeX 3$  Interfaces”. Available from CTAN, <https://www.ctan.org/pkg/l3kernel>, 2024.
- [12] The  $\LaTeX$  Project. “The  $\LaTeX 2_{\epsilon}$  sources”. Available from CTAN, <https://ctan.org/tex-archive/macros/latex/base>, 2024.
- [13] The  $\LaTeX$  Project. “ $\LaTeX$  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.
- [17] FISCHER, ULRIKE. “tagpdf –  $\LaTeX$  kernel code for PDF tagging”. Available from CTAN, <https://www.ctan.org/pkg/tagpdf>, 2024.
- [18] The  $\LaTeX$  Project. “latex-lab –  $\LaTeX$  laboratory”. Available from CTAN, <https://www.ctan.org/pkg/latex-lab>, 2024.
- [19] MITTELBAACH, FRANK. “ $\LaTeX$ ’s socket management”. Available from CTAN, <https://ctan.org/tex-archive/macros/latex/base>, 2024.

## 11 Change history

**v1.0 2024-11-01** – First public release.

12 Index of Documentation

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

|                                  |                         |                                             |                               |
|----------------------------------|-------------------------|---------------------------------------------|-------------------------------|
| C                                |                         | F                                           |                               |
| Document class:                  |                         | \footnote                                   | 5                             |
| article                          | 2                       | I                                           |                               |
| book                             | 2                       | \itemsep                                    | 8                             |
| exam                             | 2                       | K                                           |                               |
| letter                           | 2                       | Keys for \anskey provide by enumext:        |                               |
| report                           | 2                       | break-col                                   | 13                            |
| \columnbreak                     | 4, 13                   | item-join                                   | 13                            |
| \columnsep                       | 10                      | item-pos*                                   | 13                            |
| Commands provide by enumext:     |                         | item-star                                   | 13                            |
| \anskey                          | 11–14                   | item-sym*                                   | 13                            |
| \anspic                          | 11, 12, 15, 16          | Keys for \foreachkeyans provide by enumext: |                               |
| \foreachkeyans                   | 17                      | after                                       | 18                            |
| \getkeyans                       | 12, 17                  | before                                      | 18                            |
| \item*                           | 5–7, 11, 12, 14, 15     | sep                                         | 17                            |
| \item                            | 5–7, 10–12, 14, 16      | start                                       | 17, 18                        |
| \miniright                       | 11                      | step                                        | 17                            |
| \printkeyans                     | 6, 12, 18               | stop                                        | 18                            |
| \setenumextmeta                  | 6                       | wrapper                                     | 18                            |
| \setenumext                      | 5–7, 11, 12, 15, 18     | Keys for anskey* provide by enumext:        |                               |
| Counters defined by enumext:     |                         | break-col                                   | 13                            |
| enumXiii                         | 4                       | force-eol                                   | 14                            |
| enumXii                          | 4                       | item-join                                   | 13                            |
| enumXiv                          | 4                       | item-pos*                                   | 13                            |
| enumXi                           | 4                       | item-star                                   | 13                            |
| enumXviii                        | 4                       | item-sym*                                   | 13                            |
| enumXvii                         | 4                       | overwrite                                   | 14                            |
| enumXvi                          | 4                       | write-env                                   | 14                            |
| enumXv                           | 4                       | Keys for environments provide by enumext:   |                               |
| E                                |                         | above*                                      | 9                             |
| Environments provide by enumext: |                         | above                                       | 8, 9                          |
| anskey*                          | 11–14, 21               | after                                       | 10                            |
| enumext*                         | 4–11, 13–15, 18, 21, 22 | align                                       | 7, 21–23                      |
| enumext                          | 4–11, 13–16, 18, 21, 22 | base-fix                                    | 8                             |
| keyans*                          | 4–14, 22                | before*                                     | 9, 10                         |
| keyanspic                        | 4, 7, 8, 11–16, 21, 22  | before                                      | 9                             |
| keyans                           | 4–16, 22                | below*                                      | 9                             |
| Environments:                    |                         | below                                       | 9                             |
| Verbatim                         | 14                      | check-ans                                   | 12, 13                        |
| center                           | 5                       | columns-sep                                 | 4, 10, 21                     |
| description                      | 5, 22                   | columns                                     | 4, 9, 10, 21                  |
| enumerate                        | 1, 3, 5, 23             | first                                       | 10                            |
| figure                           | 5                       | font                                        | 7                             |
| flushleft                        | 5                       | item-pos*                                   | 5, 6                          |
| flushright                       | 5                       | item-sym*                                   | 5, 6                          |
| itemize                          | 5, 22                   | itemindent                                  | 8–10                          |
| list                             | 3, 5, 9, 23             | itemsep                                     | 8                             |
| minipage                         | 3–5, 8–11, 21, 23       | label-pos                                   | 16                            |
| multicols                        | 3, 4, 10, 21            | label-sep                                   | 16                            |
| quotation                        | 5                       | labelsep                                    | 3–7, 9, 10, 12, 21, 22        |
| quote                            | 5                       | labelwidth                                  | 3, 4, 6, 7, 9, 10, 12, 21, 22 |
| shortenurerate                   | 5                       | labelwith                                   | 5                             |
| tabbing                          | 5                       | label                                       | 7, 8, 10, 15, 21–23           |
| table                            | 5                       | labewdith                                   | 9                             |
| tasks                            | 5                       | layout-sep                                  | 16                            |
| trivlist                         | 5                       | layout-sty                                  | 16                            |
| verbatim                         | 5                       | layout-top                                  | 16                            |
| verse                            | 5                       | list-indent                                 | 3, 9                          |
|                                  |                         | list-offset                                 | 3, 8, 9, 22, 23               |

|                            |                      |                |                      |
|----------------------------|----------------------|----------------|----------------------|
| listparindent              | 9, 10                | \Roman*        | 7, 8                 |
| mark-ans                   | 12                   | \alph*         | 7, 8                 |
| mark-pos                   | 12                   | \arabic*       | 7, 8                 |
| mark-ref                   | 12                   | \roman*        | 7, 8                 |
| mini-env                   | 4, 9, 11, 21         | \labelsep      | 3, 7                 |
| mini-right*                | 7, 11                | \labelwidth    | 3, 7                 |
| mini-right                 | 7, 11, 21            | \linewidth     | 11                   |
| mini-sep                   | 4, 11                | \listparindent | 9                    |
| mode-box                   | 7                    |                |                      |
| no-store                   | 11–13, 22            | <b>P</b>       |                      |
| noitemsep                  | 8                    | Packages:      |                      |
| nosep                      | 8, 22                | enumerate      | 23                   |
| overwrite                  | 14                   | enumext        | 1–5, 7, 16, 21–24    |
| parsep                     | 8, 10, 16            | enumitem       | 3, 4, 23             |
| partopsep                  | 8                    | fancyvrb       | 14                   |
| ref                        | 4, 8, 21             | footnotehyper  | 5                    |
| resume*                    | 7, 10, 11            | geometry       | 21                   |
| resume                     | 7, 10, 11            | graphicx       | 21                   |
| rightmargin                | 9                    | hyperref       | 4, 5, 12, 13, 21, 23 |
| save-ans                   | 4, 6, 10–18, 21, 22  | l3keys         | 7                    |
| save-key                   | 10–12, 18            | l3prop         | 23                   |
| save-ref                   | 4, 7, 12, 13, 17, 21 | l3seq          | 23                   |
| save-sep                   | 12, 21               | luamml         | 21                   |
| series                     | 7, 10, 11            | multicol       | 1, 2, 4, 21, 23      |
| show-ans                   | 12, 22               | scontents      | 1, 2, 14, 21         |
| show-length                | 8                    | shortlst       | 5                    |
| show-pos                   | 12, 17               | tasks          | 5                    |
| start*                     | 10                   | task           | 6                    |
| start                      | 10                   | unicode-math   | 21                   |
| topsep                     | 8, 9, 16             | xsim           | 2                    |
| widest                     | 7                    | \parsep        | 8                    |
| wrap-ans                   | 12                   | \partopsep     | 8                    |
| wrap-label*                | 7, 23                |                |                      |
| wrap-label                 | 7, 21, 23            | <b>R</b>       |                      |
| wrap-opt                   | 12                   | \raggedcolumns | 4                    |
| write-env                  | 14                   | \ref           | 4                    |
|                            |                      | \rightmargin   | 9                    |
| <b>L</b>                   |                      |                |                      |
| \label                     | 4                    | <b>T</b>       |                      |
| Labels provide by enumext: |                      | \topsep        | 8                    |
| \Alph*                     | 7, 8, 15             |                |                      |

## 13 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.

### 13.1 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.

### 13.2 Initial set up

Start the DocStrip guards.

```
1 <{*package>
```

Identify the internal prefix (L<sup>A</sup>T<sub>E</sub>X3 DocStrip convention) for l3doc class.

```
2 <@@=enumext>
```

### 13.3 Declaration of the package

First we will make sure we have a minimum (super updated) version of L<sup>A</sup>T<sub>E</sub>X to work correctly.

```
3 \NeedsTeXFormat{LaTeX2e}[2024-11-01]
```

Now declare the `enumext` package.

```
4 \ProvidesExplPackage {enumext} {2024-11-01} {1.0} {Enumerate exercise sheets}
```

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

```
5 \hook_gput_code:nnn {begindocument} {enumext}
6 {
7 \IfPackageLoadedTF { multicol }
8 {
9 \msg_info:nnn { enumext } { package-load } { multicol }
10 }
11 {
12 \msg_info:nnn { enumext } { package-not-load } { multicol }
13 \RequirePackage{multicol}[2024-05-23]
14 }
15 \IfPackageLoadedTF { scontents }
16 {
17 \msg_info:nnn { enumext } { package-load } { scontents }
18 }
19 {
20 \msg_info:nnn { enumext } { package-not-load } { scontents }
21 \RequirePackage{scontents}
22 }
23 }
```

### 13.4 Definition of variables

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

```
__enumext_level_int
__enumext_level_h_int
__enumext_anskey_level_int
__enumext_keyans_level_int
__enumext_keyans_level_h_int
__enumext_keyans_pic_level_int
```

Integer variables will control the nesting levels of the environments and `\anskey` command.

```
24 \int_new:N __enumext_level_int
25 \int_new:N __enumext_level_h_int
26 \int_new:N __enumext_anskey_level_int
27 \int_new:N __enumext_keyans_level_int
28 \int_new:N __enumext_keyans_level_h_int
29 \int_new:N __enumext_keyans_pic_level_int
```

(End of definition for `\__enumext_level_int` and others.)

```

\l__enumext_starred_bool
\g__enumext_starred_bool
\l__enumext_starred_first_bool
\l__enumext_standar_bool
\g__enumext_standar_bool
\l__enumext_standar_first_bool
\l__enumext_anskey_env_bool
\l__enumext_keyans_env_bool
\g__enumext_start_line_tl
\g__enumext_envir_name_tl
\l__enumext_envir_name_tl

```

Internal variables used by functions `\__enumext_is_not_nested:`, `\__enumext_is_on_first_level:` and `\__enumext_keyans_name_and_start:` (§13.5.1).

```

30 \bool_new:N \l__enumext_starred_bool
31 \bool_new:N \g__enumext_starred_bool
32 \bool_new:N \l__enumext_starred_first_bool
33 \bool_new:N \l__enumext_standar_bool
34 \bool_new:N \g__enumext_standar_bool
35 \bool_new:N \l__enumext_standar_first_bool
36 \bool_new:N \l__enumext_anskey_env_bool
37 \bool_new:N \l__enumext_keyans_env_bool
38 \tl_new:N \g__enumext_start_line_tl
39 \tl_new:N \g__enumext_envir_name_tl
40 \tl_new:N \l__enumext_envir_name_tl

```

(End of definition for `\l__enumext_starred_bool` and others.)

```

\l__enumext_counter_i_tl
\l__enumext_counter_ii_tl
\l__enumext_counter_iii_tl
\l__enumext_counter_iv_tl
\l__enumext_counter_v_tl
\l__enumext_counter_vi_tl
\l__enumext_counter_vii_tl
\l__enumext_counter_viii_tl

```

Variables to store the “*name of the counters*” `enumXi`, `enumXii`, `enumXiii` and `enumXiv` for `enumext` environment, `enumXv` for `keyans` environment and `enumXvi` for the `keyanspic` environment. The counters `enumXvii` and `enumXviii` are used by `enumext*` and `keyans*` environments.

The initial values of these variables are set by the function `\__enumext_define_counters:Nn` (§13.11) and then modified by the function `\__enumext_label_style:Nnn` used by `label` key (§13.14).

```

41 \cs_set_protected:Npn __enumext_tmp:n #1
42 {
43 \tl_new:c { l__enumext_counter_#1_tl }
44 }
45 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { __enumext_tmp:n {#1} }

```

(End of definition for `\l__enumext_counter_i_tl` and others.)

```

\c__enumext_counter_style_tl
\l__enumext_ref_key_arg_tl
\l__enumext_ref_the_count_tl
\l__enumext_the_counter_X_tl
\l__enumext_renew_the_count_X_tl

```

Internal variables used by `ref` key (§13.14).

```

46 \tl_const:Nn \c__enumext_counter_style_tl
47 { { { arabic } { roman } { Roman } { alph } { Alph } } }
48 \tl_new:N \l__enumext_ref_key_arg_tl
49 \tl_new:N \l__enumext_ref_the_count_tl
50 \cs_set_protected:Npn __enumext_tmp:n #1
51 {
52 \tl_new:c { l__enumext_renew_the_count_#1_tl }
53 \tl_new:c { l__enumext_the_counter_#1_tl }
54 \tl_set:ce { l__enumext_the_counter_#1_tl } { \exp_not:c { theenumX#1 } }
55 }
56 \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
\g__enumext_resume_vii_int
\l__enumext_resume_name_tl
\l__enumext_resume_active_bool
\g__enumext_starred_series_tl
\g__enumext_standar_series_tl

```

Internal variables used by `resume`, `resume*` and `series` keys (§13.25).

```

57 \int_new:N \g__enumext_resume_int
58 \int_new:N \g__enumext_resume_vii_int
59 \tl_new:N \l__enumext_resume_name_tl
60 \bool_new:N \l__enumext_resume_active_bool
61 \tl_new:N \g__enumext_standar_series_tl
62 \tl_new:N \g__enumext_starred_series_tl

```

(End of definition for `\g__enumext_resume_int` and others.)

```

\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 *label style* and the variable `\g__enumext_widest_label_tl` the label width. These variables are used by `widest` (§13.15) and `label` (§13.13) keys.

```

63 \dim_new:N \l__enumext_current_widest_dim
64 \tl_new:N \g__enumext_counter_styles_tl
65 \tl_new:N \g__enumext_widest_label_tl
66 \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

```

The boolean variable `\l__enumext_leftmargin_tmp_X_bool` and the dimensional variable `\l__enumext_leftmargin_tmp_X_dim` are used by the `list-indent` key (§13.18). The variables `\l__enumext_leftmargin_X_dim` and `\l__enumext_itemindent_X_dim` are used and set by the function `\__enumext_calc_hspace:NNNNNNNNNN` (§13.38.1).

```

67 \cs_set_protected:Npn __enumext_tmp:n #1
68 {
69 \bool_new:c { \l__enumext_leftmargin_tmp_#1_bool }
70 \dim_new:c { \l__enumext_leftmargin_tmp_#1_dim }
71 \dim_new:c { \l__enumext_leftmargin_#1_dim }
72 \dim_new:c { \l__enumext_itemindent_#1_dim }
73 }
74 \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
\g__enumext_multicols_right_X_skip
\l__enumext_align_label_pos_X_str

```

Internal variables used by `columns` key (§13.22) and `align` key (§13.13).

```

75 \cs_set_protected:Npn __enumext_tmp:n #1
76 {
77 \skip_new:c { \l__enumext_multicols_above_#1_skip }
78 \skip_new:c { \l__enumext_multicols_below_#1_skip }
79 \skip_new:c { \g__enumext_multicols_right_#1_skip }
80 \str_new:c { \l__enumext_align_label_pos_#1_str }
81 }
82 \clist_map_inline:nn { i, ii, iii, iv, v } { __enumext_tmp:n {#1} }

```

(End of definition for `\l__enumext_multicols_above_X_skip` and others.)

```

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

```

Internal variables used by `\miniright` command (§13.23.4) and the keys `mini-right`, `mini-right*`, `mini-env` and `mini-sep` (§13.21, §13.23).

```

83 \int_new:N \g__enumext_minipage_stat_int
84 \skip_new:N \l__enumext_minipage_temp_skip
85 \skip_new:N \l__enumext_minipage_left_skip
86 \skip_new:N \l__enumext_minipage_right_skip
87 \skip_new:N \l__enumext_minipage_after_skip
88 \skip_new:N \g__enumext_minipage_right_skip
89 \skip_new:N \g__enumext_minipage_after_skip
90 \cs_set_protected:Npn __enumext_tmp:n #1
91 {
92 \dim_new:c { \l__enumext_minipage_left_#1_dim }
93 \bool_new:c { \l__enumext_minipage_active_#1_bool }
94 }
95 \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 (§13.13), the integer `\l__enumext_start_X_int` are used by the `start` and `start*` keys (§13.15), the token list `\l__enumext_fake_item_indent_X_tl` is used by `itemindent` key (§13.18.1), the variables `\l__enumext_label_fill_left_X_tl` and `\l__enumext_label_fill_right_X_tl` are used by the `align` key (§13.13). 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 (§13.20).

```

96 \cs_set_protected:Npn __enumext_tmp:n #1
97 {
98 \bool_new:c { \l__enumext_wrap_label_#1_bool }
99 \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 \tl_new:c { \l__enumext_label_fill_left_#1_tl }
103 \tl_new:c { \l__enumext_label_fill_right_#1_tl }
104 \bool_new:c { \l__enumext_vspace_a_star_#1_bool }
105 \bool_new:c { \l__enumext_vspace_b_star_#1_bool }
106 }
107 \clist_map_inline:nn { i, ii, iii, iv, v, vii, viii } { __enumext_tmp:n {#1} }

```

(End of definition for `\l__enumext_wrap_label_X_bool` and others.)

```

\l__enumext_store_active_bool
\l__enumext_store_name_tl
\g__enumext_store_name_tl
\l__enumext_store_anskey_arg_tl
\l__enumext_store_anskey_env_tl
\l__enumext_store_anskey_opt_tl
\l__enumext_store_current_label_tl
\l__enumext_store_current_opt_arg_tl
\l__enumext_store_current_label_tmp_tl

```

The variable `\l__enumext_store_active_bool` setting by `save-ans` key (§13.26.1) activates all the mechanism related to `\anskey`, `anskey*`, `keyans`, `keyans*` and `keyanspic` environments.

The variable `\l__enumext_store_name_tl` saves the  $\{\langle store\ name\rangle\}$  set by the `save-ans` key of the *sequence* and *prop list* in which we will store, the variable `\g__enumext_store_name_tl` it's just a global copy of  $\{\langle store\ name\rangle\}$  used by different functions.

The variable `\l__enumext_store_anskey_arg_tl` save the *argument* of `\anskey` (§13.30) and the variables `\l__enumext_store_anskey_env_tl` and `\l__enumext_store_anskey_opt_tl` save the  $\langle body\rangle$  and the  $\langle keys\rangle$  of the environment `anskey*` (§13.31).

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*` (§13.37) and `\anspic*` (§13.42.2) 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.

```

108 \bool_new:N \l__enumext_store_active_bool
109 \tl_new:N \l__enumext_store_name_tl
110 \tl_new:N \g__enumext_store_name_tl
111 \tl_new:N \l__enumext_store_anskey_arg_tl
112 \tl_new:N \l__enumext_store_anskey_env_tl
113 \tl_new:N \l__enumext_store_anskey_opt_tl
114 \tl_new:N \l__enumext_store_current_label_tl
115 \tl_new:N \l__enumext_store_current_opt_arg_tl
116 \tl_new:N \l__enumext_store_current_label_tmp_tl

```

(End of definition for `\l__enumext_store_active_bool` and others.)

```

\l__enumext_setkey_tmpa_tl
\l__enumext_setkey_tmpb_tl
\l__enumext_setkey_tmpa_int
\l__enumext_setkey_tmpa_seq
\l__enumext_setkey_tmpb_seq

```

Internal variables used by the command `\setenumext` (§13.48).

```

117 \tl_new:N \l__enumext_setkey_tmpa_tl
118 \tl_new:N \l__enumext_setkey_tmpb_tl
119 \int_new:N \l__enumext_setkey_tmpa_int
120 \seq_new:N \l__enumext_setkey_tmpa_seq
121 \seq_new:N \l__enumext_setkey_tmpb_seq

```

(End of definition for `\l__enumext_setkey_tmpa_tl` and others.)

```

\l__enumext_meta_path_tl
\l__enumext_foreach_print_seq
\l__enumext_foreach_name_prop_tl
\g__enumext_foreach_default_keys_tl

```

Internal variables used by the `\printkeyans` command (§13.47) and `\foreachkeyans` command (§13.50).

```

122 \tl_new:N \l__enumext_meta_path_tl
123 \seq_new:N \l__enumext_foreach_print_seq
124 \tl_new:N \l__enumext_foreach_name_prop_tl
125 \tl_new:N \g__enumext_foreach_default_keys_tl

```

(End of definition for `\l__enumext_meta_path_tl` and others.)

```

\l__enumext_print_keyans_starred_tl
\l__enumext_print_keyans_star_bool
\l__enumext_mark_position_str
\g__enumext_item_symbol_aux_tl
\l__enumext_print_keyans_X_tl
\l__enumext_store_save_key_X_tl
\l__enumext_store_save_key_X_bool
\l__enumext_store_upper_level_X_bool

```

Internal variables used by command `\printkeyans` (§13.47), `show-pos` key (§13.27), `item-sym*` key (§13.35), `save-key` key (§13.27.2) and “*storing structure*”.

```

126 \tl_new:N \l__enumext_print_keyans_starred_tl
127 \bool_new:N \l__enumext_print_keyans_star_bool
128 \str_new:N \l__enumext_mark_position_str
129 \tl_new:N \g__enumext_item_symbol_aux_tl
130 \cs_set_protected:Npn __enumext_tmp:n #1
131 {
132 \tl_new:c { \l__enumext_print_keyans_#1_tl }
133 \tl_new:c { \l__enumext_store_save_key_#1_tl }
134 \bool_new:c { \l__enumext_store_save_key_#1_bool }
135 \bool_new:c { \l__enumext_store_upper_level_#1_bool }
136 }
137 \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.)

```

\l__enumext_anspic_args_seq
\l__enumext_anspic_mini_width_dim
\l__enumext_anspic_above_int
\l__enumext_anspic_below_int
\l__enumext_anspic_label_above_bool
\l__enumext_anspic_mini_pos_str
\g__enumext_keyans_pic_parsep_skip
\l__enumext_anspic_label_box
\l__enumext_anspic_body_box
\l__enumext_anspic_label_htdp_dim
\l__enumext_anspic_body_htdp_dim

```

Internal variables used by `keyanspic` environment and `\anspic` command (§13.42.1).

```

138 \seq_new:N \l__enumext_anspic_args_seq
139 \dim_new:N \l__enumext_anspic_mini_width_dim
140 \int_new:N \l__enumext_anspic_above_int
141 \int_new:N \l__enumext_anspic_below_int
142 \bool_new:N \l__enumext_anspic_label_above_bool
143 \str_new:N \l__enumext_anspic_mini_pos_str
144 \skip_new:N \g__enumext_keyans_pic_parsep_skip
145 \box_new:N \l__enumext_anspic_label_box
146 \box_new:N \l__enumext_anspic_body_box
147 \dim_new:N \l__enumext_anspic_label_htdp_dim
148 \dim_new:N \l__enumext_anspic_body_htdp_dim

```

(End of definition for `\l__enumext_anspic_args_seq` and others.)

```
\l__enumext_check_answers_bool
\g__enumext_check_ans_key_bool
\l__enumext_check_start_line_env_tl
\l__enumext_item_star_exec_int
\l__enumext_item_star_wrap_bool
\g__enumext_check_starred_cmd_int
\g__enumext_item_anskey_int
\g__enumext_item_number_int
\g__enumext_item_number_bool
\g__enumext_item_answer_diff_int
```

Internal variables used by “*internal check answer*” mechanism (§13.26.3) used by the `check-ans`, `no-store`, `wrap-key` keys and check for starred commands `\item*` in `keyans` and `keyans*` environments and `\anspic*` in `keyanspic` environment.

```
149 \bool_new:N \l__enumext_check_answers_bool
150 \bool_new:N \g__enumext_check_ans_key_bool
151 \tl_new:N \l__enumext_check_start_line_env_tl
152 \int_new:N \l__enumext_item_star_exec_int
153 \bool_new:N \l__enumext_item_star_wrap_bool
154 \int_new:N \g__enumext_check_starred_cmd_int
155 \int_new:N \g__enumext_item_anskey_int
156 \int_new:N \g__enumext_item_number_int
157 \bool_new:N \l__enumext_item_number_bool
158 \int_new:N \g__enumext_item_answer_diff_int
```

(End of definition for `\l__enumext_check_answers_bool` and others.)

```
\l__enumext_hyperref_bool
\l__enumext_footnotes_key_bool
```

The boolean variable `\l__enumext_hyperref_bool` will determine if the `hyperref` package is present or load in memory (§13.7). The boolean variable `\l__enumext_footnotes_key_bool` determine if `hyperref` is load with key `hyperfootnotes=true`.

```
159 \bool_new:N \l__enumext_hyperref_bool
160 \bool_new:N \l__enumext_footnotes_key_bool
```

(End of definition for `\l__enumext_hyperref_bool` and `\l__enumext_footnotes_key_bool`.)

```
\l__enumext_newlabel_arg_one_tl
\l__enumext_newlabel_arg_two_tl
\l__enumext_write_aux_file_tl
\l__enumext_label_copy_X_tl
```

Internal variables used by `save-ref` key (§13.27). The variables `\l__enumext_label_copy_X_tl` correspond to temporary copies of the  $\langle labels \rangle$  defined by level on which operations will be performed.

The variables `\l__enumext_newlabel_arg_one_tl` and `\l__enumext_newlabel_arg_two_tl` will be used to form the arguments passed to the function `\__enumext_newlabel:nn` (§13.7) and the variable `\l__enumext_write_aux_file_tl` will be in charge of executing the writing code in the `.aux` file.

```
161 \tl_new:N \l__enumext_newlabel_arg_one_tl
162 \tl_new:N \l__enumext_newlabel_arg_two_tl
163 \tl_new:N \l__enumext_write_aux_file_tl
164 \cs_set_protected:Npn __enumext_tmp:n #1
165 {
166 \tl_new:c { l__enumext_label_copy_#1_tl }
167 }
168 \clist_map_inline:nn { i, ii, iii, iv, v, vi, vii, viii } { __enumext_tmp:n {#1} }
```

(End of definition for `\l__enumext_newlabel_arg_one_tl` and others.)

```
\g__enumext_footnote_standar_int
\g__enumext_footnote_starred_int
\g__enumext_footnote_standar_arg_seq
\g__enumext_footnote_starred_arg_seq
\g__enumext_footnote_standar_int_seq
\g__enumext_footnote_starred_int_seq
```

Internal variables used for redefinition of `\footnote` (§13.8).

```
169 \int_new:N \g__enumext_footnote_standar_int
170 \int_new:N \g__enumext_footnote_starred_int
171 \seq_new:N \g__enumext_footnote_standar_arg_seq
172 \seq_new:N \g__enumext_footnote_starred_arg_seq
173 \seq_new:N \g__enumext_footnote_standar_int_seq
174 \seq_new:N \g__enumext_footnote_starred_int_seq
```

(End of definition for `\g__enumext_footnote_standar_int` and others.)

```
\l__enumext_item_starred_X_bool
\l__enumext_item_column_pos_X_int
\g__enumext_item_count_all_X_int
\l__enumext_joined_item_X_int
\l__enumext_joined_item_aux_X_int
\l__enumext_tmpa_X_int
\l__enumext_tmpa_X_dim
\l__enumext_item_text_X_box
\l__enumext_joined_width_X_dim
\l__enumext_item_width_X_dim
\g__enumext_item_symbol_aux_X_tl
\l__enumext_align_label_X_str
\g__enumext_minipage_active_X_bool
\l__enumext_miniright_code_X_box
\g__enumext_minipage_center_X_bool
\g__enumext_minipage_right_X_dim
\g__enumext_minipage_right_X_skip
```

Internal variables used by `enumext*` and `keyans*` environments.

```
175 \cs_set_protected:Npn __enumext_tmp:n #1
176 {
177 \bool_new:c { l__enumext_item_starred_#1_bool }
178 \int_new:c { l__enumext_item_column_pos_#1_int }
179 \int_new:c { g__enumext_item_count_all_#1_int }
180 \int_new:c { l__enumext_joined_item_#1_int }
181 \int_new:c { l__enumext_joined_item_aux_#1_int }
182 \int_new:c { l__enumext_tmpa_#1_int }
183 \dim_new:c { l__enumext_tmpa_#1_dim }
184 \box_new:c { l__enumext_item_text_#1_box }
185 \dim_new:c { l__enumext_joined_width_#1_dim }
186 \dim_new:c { l__enumext_item_width_#1_dim }
187 \tl_new:c { g__enumext_item_symbol_aux_#1_tl }
188 \str_new:c { l__enumext_align_label_#1_str }
189 \bool_new:c { g__enumext_minipage_active_#1_bool }
190 \box_new:c { l__enumext_miniright_code_#1_box }
191 \bool_new:c { g__enumext_minipage_center_#1_bool }
```

```

192 \dim_new:c { g__enumext_minipage_right_#1_dim }
193 \skip_new:c { g__enumext_minipage_right_#1_skip }
194 }
195 \clist_map_inline:nn { vii, viii } { __enumext_tmp:n {#1} }

```

(End of definition for `\__enumext_item_starred_X_bool` and others.)

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

```

196 \clist_const:Nn \c__enumext_all_envs_clist
197 {
198 {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv},
199 {keyans}{v}, {enumext*}{vii}, {keyans*}{viii}
200 }

```

(End of definition for `\c__enumext_all_envs_clist`.)

### 13.5 Some utility functions

`\keys_precompile:neN` `\seq_use:NV` Non-standard kernel variants used by the `\printkeyans` command (§13.47) and `\foreachkeyans` command (§13.50).

```

201 \cs_generate_variant:Nn \keys_precompile:nnN { neN }
202 \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.

```

203 \cs_new_protected:Npn __enumext_at_begin_document:n #1
204 {
205 \hook_gput_code:nnn {begindocument} {enumext} { #1 }
206 }

```

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

`\__enumext_after_env:nn` `\__enumext_before_env:nn` A internal “hook” functions for execute code `mini-right` and `mini-right*` keys outside the `enumext*` and `keyans*` environments and print check-ans outside the `enumext` and `enumext*` environments.

```

207 \cs_new_protected:Npn __enumext_after_env:nn #1 #2
208 {
209 \hook_gput_code:nnn {env/#1/after} {enumext} {#2}
210 }
211 \cs_new_protected:Npn __enumext_before_env:nn #1 #2
212 {
213 \hook_gput_code:nnn {env/#1/before} {enumext} {#2}
214 }

```

(End of definition for `\__enumext_after_env:nn` and `\__enumext_before_env:nn`.)

`\__enumext_level:` Function for check current level in `enumext`.

```

215 \cs_new:Nn __enumext_level:
216 {
217 \int_to_roman:n { __enumext_level_int }
218 }

```

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

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

```

219 \prg_new_protected_conditional:Npnn __enumext_if_is_int:n #1 { T, F, TF }
220 {
221 \regex_match:nnTF { ^[\+|-]?[\d]+$ } {#1} % $
222 { \prg_return_true: }
223 { \prg_return_false: }
224 }

```

(End of definition for `\__enumext_if_is_int:nT`, `\__enumext_if_is_int:nF`, and `\__enumext_if_is_int:nTF`.)

`\__enumext_regex_counter_style:`

The internal function `\__enumext_regex_counter_style:` replace the ‘`*`’ with the actual counter of the running level and is used by the `ref` key. It loops through the defined counter styles in `\c__enumext_counter_style_tl` and replace ‘`*`’ by real command, for example, looking for `\arabic*` and replacing that by `\arabic{<counter>}` defined on the current level.

```

225 \cs_new_protected:Nn __enumext_regex_counter_style:
226 {
227 \tl_map_inline:Nn \c__enumext_counter_style_tl
228 {
229 \regex_replace_once:nnN { \c{##1}* }
230 { \c{##1}\cB{\u{\l__enumext_ref_the_count_tl}\cE} } \l__enumext_ref_key_arg_tl
231 }
232 }

```

(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.

```

233 \cs_new:Npn __enumext_show_length:nnn #1 #2 #3
234 {
235 * ~ #2
236 \prg_replicate:nn { 14 - \str_count:n {#2} } { ~ }
237 = ~ \use:c { #1_use:c } { \l__enumext_#2_#3_#1 } \\
238 }

```

(End of definition for `\__enumext_show_length:nnn.`)

`\__enumext_unskip_unkern:`

The function `\__enumext_unskip_unkern:` will remove the last `<skip>` or `<kern>` at execution time using the values `11` and `12` of `\lastnodetype` to apply `\unskip` or `\unkern` according to the case.

```

239 \cs_new_protected:Nn __enumext_unskip_unkern:
240 {
241 \int_case:nnT { \lastnodetype }
242 {
243 { 11 } { \unskip }
244 { 12 } { \unkern }
245 }
246 }

```

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

### 13.5.1 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 NOT nested in each other and save the environment name in `\l__enumext_envir_name_tl`.

```

247 \cs_new_protected:Nn __enumext_is_not_nested:
248 {
249 \str_case:en { \@currentenv }
250 {
251 {enumext}
252 {
253 \tl_set:Nn \l__enumext_envir_name_tl { enumext }
254 \bool_lazy_and:nnT
255 { \bool_not_p:n { \g__enumext_standar_bool } }
256 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
257 {
258 \bool_gset_true:N \g__enumext_standar_bool
259 }
260 }
261 {enumext*}
262 {
263 \tl_set:Nn \l__enumext_envir_name_tl { enumext* }
264 \bool_lazy_and:nnT
265 { \bool_not_p:n { \g__enumext_starred_bool } }
266 { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
267 {
268 \bool_gset_true:N \g__enumext_starred_bool
269 }
270 }
271 }
272 }

```

The function `\__enumext_is_on_first_level:` will set the variables `\l__enumext_standar_first_bool` (§13.26.1), `\l__enumext_starred_first_bool` (§13.26.1) and `\l__enumext_anskey_env_bool` (§13.31) 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.

```

273 \cs_new_protected:Nn __enumext_is_on_first_level:
274 {
275 \bool_lazy_all:nT
276 {
277 { \bool_if_p:N \g__enumext_standar_bool }
278 { \int_compare_p:nNn { \l__enumext_level_int } = { 1 } }
279 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
280 }
281 {
282 \bool_set_true:N \l__enumext_standar_first_bool
283 \bool_set_true:N \l__enumext_anskey_env_bool
284 \tl_gset:Nn \g__enumext_envir_name_tl { enumext }
285 \tl_gset:Ne \g__enumext_start_line_tl
286 {
287 on ~ line ~ \exp_not:V \inputlineno
288 }
289 }
290 \bool_lazy_all:nT
291 {
292 { \bool_if_p:N \g__enumext_starred_bool }
293 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
294 { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
295 }
296 {
297 \bool_set_true:N \l__enumext_starred_first_bool
298 \bool_set_true:N \l__enumext_anskey_env_bool
299 \tl_gset:Nn \g__enumext_envir_name_tl { enumext* }
300 \tl_gset:Ne \g__enumext_start_line_tl
301 {
302 on ~ line ~ \exp_not:V \inputlineno
303 }
304 }
305 }

```

(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.

```

306 \cs_new_protected:Nn __enumext_keyans_name_and_start:
307 {
308 \str_case:en { \@currenvir }
309 {
310 {keyans}
311 {
312 \tl_set:Nn \l__enumext_envir_name_tl { keyans }
313 \tl_set:Ne \l__enumext_check_start_line_env_tl
314 {
315 in ~ 'keyans' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
316 }
317 }
318 {keyans*}
319 {
320 \tl_set:Nn \l__enumext_envir_name_tl { keyans* }
321 \tl_set:Ne \l__enumext_check_start_line_env_tl
322 {
323 in ~ 'keyans*' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
324 }
325 }
326 {keyanspic}
327 {
328 \tl_set:Nn \l__enumext_envir_name_tl { keyanspic }
329 \tl_set:Ne \l__enumext_check_start_line_env_tl
330 {

```



```

331 in ~ 'keyanspic' ~ start ~ on ~ line ~ \exp_not:V \inputlineno
332 }
333 }
334 }
335 }

```

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

### 13.5.2 Utilities for log and terminal

The function `\__enumext_reset_global_vars:` will be passed to the function `\__enumext_execute_after_env:` and will return the global variables to their default values after being used.

```

336 \cs_new_protected:Nn __enumext_reset_global_vars:
337 {
338 __enumext_reset_global_int:
339 __enumext_reset_global_bool:
340 __enumext_reset_global_tl:
341 }
342 \cs_new_protected:Nn __enumext_reset_global_int:
343 {
344 \int_gzero:N \g__enumext_item_number_int
345 \int_gzero:N \g__enumext_item_anskey_int
346 \int_gzero:N \g__enumext_item_answer_diff_int
347 }
348 \cs_new_protected:Nn __enumext_reset_global_bool:
349 {
350 \bool_gset_false:N \g__enumext_check_ans_key_bool
351 \bool_gset_false:N \g__enumext_standar_bool
352 \bool_gset_false:N \g__enumext_starred_bool
353 }
354 \cs_new_protected:Nn __enumext_reset_global_tl:
355 {
356 \tl_gclear:N \g__enumext_store_name_tl
357 \tl_gclear:N \g__enumext_start_line_tl
358 \tl_gclear:N \g__enumext_envir_name_tl
359 }

```

(End of definition for `\__enumext_reset_global_vars:` and others.)

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 *prop list* and *sequence* created by the *save-ans* key along with the value of the integer variable created for the *resume* key.

```

360 \cs_new_protected:Nn __enumext_log_global_vars:
361 {
362 \msg_log:nneeee { enumext } { prop-seq-int-hook }
363 { \g__enumext_store_name_tl }
364 { \prop_count:c { g__enumext_ \g__enumext_store_name_tl _prop } }
365 { \seq_count:c { g__enumext_ \g__enumext_store_name_tl _seq } }
366 { \int_use:c { g__enumext_resume_ \g__enumext_store_name_tl _int } }
367 }

```

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.

```

368 \cs_new_protected:Nn __enumext_log_answer_vars:
369 {
370 \msg_log:nneee { enumext } { item-answer-hook }
371 { \int_use:N \g__enumext_item_number_int }
372 { \int_use:N \g__enumext_item_anskey_int }
373 { \int_eval:n { \g__enumext_item_number_int - \g__enumext_item_anskey_int } }
374 }

```

(End of definition for `\__enumext_log_global_vars:` and `\__enumext_log_answer_vars:`.)

## 13.6 Copying list and minipage environments

The `list` environment provided by  $\text{\LaTeX}$  has the following plain form:

```

\list{⟨arg one⟩}{⟨arg two⟩}
 \item[⟨opt⟩]
\endlist

```

And `minipage` environment provided by  $\text{\LaTeX}$  has the following (simplified) plain form:

```

\minipage[⟨pos⟩][⟨height⟩][⟨inner-pos⟩]{⟨width⟩}
 ⟨internal implement⟩
\endminipage

```

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

- For compatibility with *tagged* PDF we should use `\NewCommandCopy` and not `\cs_new_eq:NN` for `\item`. When *tagged* PDF is active `\item` is redefined using `\ltxcmd` (see `latex-lab-block[18]`).

```

__enumext_start_list:nn
__enumext_stop_list:
__enumext_item_std:w
__enumext_minipage:w
__enumext_endminipage:

```

The functions `\__enumext_start_list:nn` and `\__enumext_stop_list:` correspond to copies of `\list` and `\endlist` from plain definition of `\list` environment, the function `\__enumext_item_std:w` is a copy of the `\item` command.

```

375 __enumext_at_begin_document:n
376 {
377 \cs_new_eq:NN __enumext_start_list:nn \list
378 \cs_new_eq:NN __enumext_stop_list: \endlist
379 \NewCommandCopy __enumext_item_std:w \item
380 }

```

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

```

381 __enumext_at_begin_document:n
382 {
383 \cs_new_eq:NN __enumext_minipage:w \minipage
384 \cs_new_eq:NN __enumext_endminipage: \endminipage
385 }

```

(End of definition for `\__enumext_start_list:nn` and others.)

### 13.7 Compatibility with hyperref and footnotehyper

```

__enumext_after_hyperref:
__enumext_hypertarget:nn
__enumext_phantomsection:

```

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

```

386 \hook_gput_code:nnn { begindocument } { enumext } { __enumext_after_hyperref: }
387 \hook_gset_rule:nnnn { begindocument } { enumext } { after } { hyperref }

```

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”.

```

388 \cs_new_protected:Nn __enumext_after_hyperref:
389 {
390 \IfPackageLoadedTF { hyperref }
391 {
392 \msg_info:nnn { enumext } { package-load } { hyperref }
393 \bool_set_true:N \l__enumext_hyperref_bool
394 \IfHyperBoolean{hyperfootnotes}
395 {
396 \bool_set_true:N \l__enumext_footnotes_key_bool
397 }
398 { }
399 }
400 { }

```

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`.

```

401 \bool_if:NT \l__enumext_footnotes_key_bool
402 {
403 \IfPackageLoadedTF { footnotehyper }
404 {
405 \msg_info:nnn { enumext } { package-load } { footnotehyper }
406 }
407 {
408 \bool_set_false:N \l__enumext_footnotes_key_bool
409 }
410 }

```

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.

```

411 \bool_if:NTF \l__enumext_hyperref_bool
412 {

```

```

413 \cs_new_eq:NN __enumext_hypertarget:nn \hypertarget
414 \cs_new_eq:NN __enumext_phantomsection: \phantomsection
415 }
416 {
417 \cs_new_eq:NN __enumext_hypertarget:nn \use_none:nn
418 \cs_new_eq:NN __enumext_phantomsection: \prg_do_nothing:
419 }
420 }

```

(End of definition for `\__enumext_after_hyperref:`, `\__enumext_hypertarget:nn`, and `\__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.

```

421 \cs_new_protected:Npn __enumext_newlabel:nn #1 #2
422 {
423 \protected@write \@auxout { }
424 {
425 \token_to_str:N \newlabel {#1}
426 {
427 {#2}
428 \bool_if:NT \l__enumext_hyperref_bool
429 { { \thepage } {#2} {#1} }
430 { }
431 }
432 }
433 __enumext_hypertarget:nn {#1} { }
434 __enumext_phantomsection:
435 }

```

(End of definition for `\__enumext_newlabel:nn`.)

### 13.8 Internal redefining `\footnote` command

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

`\__enumext_footnotetext:nn` `\__enumext_renew_footnote:` `\__enumext_print_footnote:` `\__enumext_renew_footnote_mini:` `\__enumext_print_footnote_mini:` Redefinition of the `\footnote` command using `\footnotetext` and `\footnotemark` for the `mini-env` key in the `enumext` and `keyans` environments.

```

436 \cs_new_protected:Nn __enumext_footnotetext:nn
437 {
438 \footnotetext[#1]{#2}
439 }
440 \cs_new_protected:Nn __enumext_renew_footnote:
441 {
442 \RenewDocumentCommand \footnote { o +m }
443 {
444 \tl_if_novalue:nTF {##1}
445 {
446 \stepcounter{footnote}
447 \int_gset_eq:Nc \g__enumext_footnote_standar_int { c@footnote }
448 }
449 {
450 \int_gset:Nn \g__enumext_footnote_standar_int { ##1 }
451 }
452 \footnotemark [\g__enumext_footnote_standar_int]
453 \seq_gput_right:Nn \g__enumext_footnote_standar_arg_seq { ##2 }
454 \seq_gput_right:NV
455 \g__enumext_footnote_standar_int_seq \g__enumext_footnote_standar_int
456 }
457 }
458 \cs_new_protected:Nn __enumext_print_footnote:
459 {
460 \seq_if_empty:NF \g__enumext_footnote_standar_int_seq
461 {
462 \seq_map_pairwise_function:NNN

```

```

463 \g__enumext_footnote_standar_int_seq
464 \g__enumext_footnote_standar_arg_seq
465 __enumext_footnotetext:nn
466 }
467 \seq_gclear:N \g__enumext_footnote_standar_arg_seq
468 \seq_gclear:N \g__enumext_footnote_standar_int_seq
469 }

```

The `enumext*` and `keyans*` environments are implemented using `minipage` so we must also redefine `\footnote` to keep these numbering as if it were part of the document.

```

470 \cs_new_protected:Nn __enumext_renew_footnote_mini:
471 {
472 \RenewDocumentCommand \footnote { o +m }
473 {
474 \tl_if_novalue:nTF {##1}
475 {
476 \stepcounter{footnote}
477 \int_gset_eq:Nc \g__enumext_footnote_starred_int { c@footnote }
478 }
479 {
480 \int_gset:Nn \g__enumext_footnote_starred_int { ##1 }
481 }
482 \footnotemark [\g__enumext_footnote_starred_int]
483 \seq_gput_right:Nn \g__enumext_footnote_starred_arg_seq { ##2 }
484 \seq_gput_right:NV
485 \g__enumext_footnote_starred_int_seq \g__enumext_footnote_starred_int
486 }
487 }
488 \cs_new_protected:Nn __enumext_print_footnote_mini:
489 {
490 \seq_if_empty:NF \g__enumext_footnote_starred_int_seq
491 {
492 \seq_map_pairwise_function:NNN
493 \g__enumext_footnote_starred_int_seq
494 \g__enumext_footnote_starred_arg_seq
495 __enumext_footnotetext:nn
496 }
497 \seq_gclear:N \g__enumext_footnote_starred_arg_seq
498 \seq_gclear:N \g__enumext_footnote_starred_int_seq
499 }

```

(End of definition for `\__enumext_footnotetext:nn` and others.)

```

__enumext_renew_footnote_standar:
__enumext_print_footnote_standar:
__enumext_renew_footnote_starred:
__enumext_print_footnote_starred:

```

We encapsulate the redefinition of `\footnote` to pass it to internal `__enumext_mini_page` environment used by the `mini-env` key in the `enumext` and `keyans` environments. We will run the redefinition when `tagged` PDF is active or when the `footnotehyper` package is not loaded.

```

500 \cs_new_protected:Nn __enumext_renew_footnote_standar:
501 {
502 \bool_if:NT \g__enumext_standar_bool
503 {
504 \IfDocumentMetadataTF
505 {
506 __enumext_renew_footnote:
507 }
508 {
509 \bool_if:NF \l__enumext_footnotes_key_bool
510 {
511 __enumext_renew_footnote:
512 }
513 }
514 }
515 }
516 \cs_new_protected:Nn __enumext_print_footnote_standar:
517 {
518 \bool_if:NT \g__enumext_standar_bool
519 {
520 \IfDocumentMetadataTF
521 {
522 __enumext_print_footnote:
523 }
524 {

```

```

525 \bool_if:NF \l__enumext_footnotes_key_bool
526 {
527 __enumext_print_footnote:
528 }
529 }
530 }
531 }

```

We encapsulate the redefinition of `\footnote` to pass it to the `enumext*` and `keyans*` environments. We will run the redefinition when *tagged* PDF is active or when the `footnotehyper` package is not loaded.

```

532 \cs_new_protected:Nn __enumext_renew_footnote_starred:
533 {
534 \IfDocumentMetadataTF
535 {
536 __enumext_renew_footnote_mini:
537 }
538 {
539 \bool_if:NF \l__enumext_footnotes_key_bool
540 {
541 __enumext_renew_footnote_mini:
542 }
543 }
544 }
545 \cs_new_protected:Nn __enumext_print_footnote_starred:
546 {
547 \IfDocumentMetadataTF
548 {
549 __enumext_print_footnote_mini:
550 }
551 {
552 \bool_if:NF \l__enumext_footnotes_key_bool
553 {
554 __enumext_print_footnote_mini:
555 }
556 }
557 }

```

In `enumext*` and `keyans*` environments we need to use “hooks” to print `\footnote` with support for *tagged* PDF.

```

558 __enumext_after_env:nn { enumext* }
559 {
560 __enumext_print_footnote_starred:
561 }
562 __enumext_after_env:nn { keyans* }
563 {
564 __enumext_print_footnote_starred:
565 }

```

(End of definition for `\__enumext_renew_footnote_standar:` and others.)

### 13.9 The internal minipage environment

```

__enumext_internal_mini_page:
__enumext_mini_env*

```

The function `\__enumext_internal_mini_page:` creates a internal `\__enumext_mini_page` 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. Within this environment we redefine `\footnote` to make them look the same as if they were elsewhere in the document. This function is passed to the function `\__enumext_safe_exec:` in the `enumext` environment definition (§13.39) and `\__enumext_safe_exec_vii:` in the `enumext*` environment definition (§13.44)

```

566 \cs_new_protected:Nn __enumext_internal_mini_page:
567 {
568 \int_compare:nNtT { \l__enumext_level_int } = { 0 }
569 {
570 \DeclareDocumentEnvironment{__enumext_mini_page}{ m }
571 {
572 __enumext_renew_footnote_standar:
573 __enumext_minipage:w [t] { ##1 }
574 \legacy_if_gset_false:n { @minipage }
575 \skip_vertical:N \c_zero_skip
576 }
577 }

```

```

577 {
578 \skip_vertical:N \c_zero_skip
579 __enumext_endminipage:
580 __enumext_print_footnote_standar:
581 }
582 }
583 }

```

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

### 13.10 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.

```

584 \dim_zero_new:N \itemwidth

```

### 13.11 Definition of counters

```

__enumext_define_counters:Nn
enumXi
enumXii
enumXiii
enumXiv
enumXv
enumXvi
enumXvii
enumXviii

```

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 `\__enumext_counter_X_tl` for “store” the counter’s name.
- #2: The counter’s name.

```

585 \cs_new_protected:Npn __enumext_define_counters:Nn #1 #2
586 {
587 \cs_if_exist:cTF { c@ #2 }
588 { \msg_fatal:nnn { enumext } { counters } { #2 } }
589 {
590 \tl_set:Nn #1 { #2 }
591 \newcounter { #2 }
592 }
593 }

```

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

```

594 __enumext_define_counters:Nn __enumext_counter_i_tl { enumXi }
595 __enumext_define_counters:Nn __enumext_counter_ii_tl { enumXii }
596 __enumext_define_counters:Nn __enumext_counter_iii_tl { enumXiii }
597 __enumext_define_counters:Nn __enumext_counter_iv_tl { enumXiv }
598 __enumext_define_counters:Nn __enumext_counter_v_tl { enumXv }
599 __enumext_define_counters:Nn __enumext_counter_vi_tl { enumXvi }
600 __enumext_define_counters:Nn __enumext_counter_vii_tl { enumXvii }
601 __enumext_define_counters:Nn __enumext_counter_viii_tl { enumXviii }

```

(End of definition for `\__enumext_define_counters:Nn` and others.)

### 13.12 Definition of labels

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

```

__enumext_register_counter_style:Nn

```

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

```

602 \cs_new_protected:Npn __enumext_register_counter_style:Nn #1 #2
603 {
604 \tl_const:cn { c__enumext_widest_ \cs_to_str:N #1 _tl } {#2}
605 \tl_gput_right:Nn \g__enumext_counter_styles_tl {#1}
606 }
607 __enumext_register_counter_style:Nn \arabic { 0 }
608 __enumext_register_counter_style:Nn \Alph { M }
609 __enumext_register_counter_style:Nn \alph { m }
610 __enumext_register_counter_style:Nn \Roman { VIII }
611 __enumext_register_counter_style:Nn \roman { viii }

```

(End of definition for `\__enumext_register_counter_style:Nn`.)



```

__enumext_label_width_by_box:Nn
__enumext_label_width_by_box:cv

```

The function `\__enumext_label_width_by_box:Nn` set the default `\labelwidth` using a box width if no `\labelwidth` key is passed.

```

612 \cs_new_protected:Npn __enumext_label_width_by_box:Nn #1 #2
613 {
614 \hbox_set:Nn \l__enumext_label_width_by_box {#2}
615 \dim_set:Nn #1 { \box_wd:N \l__enumext_label_width_by_box }
616 }
617 \cs_generate_variant:Nn __enumext_label_width_by_box:Nn { cv }

```

(End of definition for `\__enumext_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 `\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{<counter>}`, and doing the same for the `\g__enumext_widest_label_tl` to keep both in sync.

```

618 \cs_new_protected:Npn __enumext_label_style:Nnn #1 #2 #3
619 {
620 \tl_clear_new:N #1
621 \tl_put_right:Ne #1 { \tl_trim_spaces:n {#3} }
622 \tl_gset_eq:NN \g__enumext_widest_label_tl #1
623 \tl_map_inline:Nn \g__enumext_counter_styles_tl
624 {
625 \tl_replace_all:Nne #1 { ##1* } { \exp_not:N ##1 {#2} }
626 \tl_greplace_all:Nne \g__enumext_widest_label_tl { ##1* }
627 { \tl_use:c { c__enumext_widest_ \cs_to_str:N ##1 _tl } }
628 }
629 __enumext_label_width_by_box:Nn \l__enumext_current_widest_dim
630 { \tl_use:N \g__enumext_widest_label_tl }
631 \tl_set_eq:cN { the #2 } #1
632 }
633 \cs_generate_variant:Nn __enumext_label_style:Nnn { cvn }

```

(End of definition for `\__enumext_label_style:Nnn`.)

### 13.13 Setting keys associated with label

When `tagged PDF` is active `\makelabel` is redefined using `\makebox` to work correctly (§13.34). From the user side it is convenient to have a key that allows using this redefinition with `\makebox` without having `\IfDocumentMetadataTF` active.

mode-box

We define the key `mode-box` only for the “first level” of `enumext` and `enumext*` environments.

```

634 \cs_set_protected:Npn __enumext_tmp:n #1
635 {
636 \keys_define:nn { enumext / #1 }
637 {
638 mode-box .bool_set:N = \l__enumext_mode_box_bool,
639 mode-box .initial:n = false,
640 mode-box .value_forbidden:n = true,
641 }
642 }
643 \clist_map_inline:nn { level-1, enumext* } { __enumext_tmp:n {#1} }

```

(End of definition for `mode-box`.)

font  
labelsep  
labelwidth  
wrap-label  
wrap-label\*

Definition of keys `font`, `labelsep`, `labelwidth`, `wrap-label` and `wrap-label*` keys for `enumext` and `keyans` environments.

```

644 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
645 {
646 \keys_define:nn { enumext / #1 }
647 {
648 font .tl_set:c = { \l__enumext_label_font_style_#2_tl },
649 font .value_required:n = true,
650 labelsep .dim_set:c = { \l__enumext_labelsep_#2_dim },
651 labelsep .initial:n = {0.3333em},
652 labelsep .value_required:n = true,
653 labelwidth .dim_set:c = { \l__enumext_labelwidth_#2_dim },
654 labelwidth .value_required:n = true,
655 wrap-label .cs_set_protected:cp = { __enumext_wrapper_label_#2:n } ##1,
656 wrap-label .initial:n = {##1},
657 wrap-label .value_required:n = true,

```

```

658 wrap-label* .code:n = {
659 \bool_set_true:c { l__enumext_wrap_label_opt_#2_bool }
660 \keys_set:nn { enumext / #1 } { wrap-label = {##1} }
661 },
662 wrap-label* .value_required:n = true,
663 }
664 }
665 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

```

(End of definition for font and others.)

`align` The `align` key is implemented differently for “starred” and “non starred” environments. For compatibility with tagged PDF we must set `\l__enumext_align_label_pos_X_str`.

```

666 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
667 {
668 \keys_define:nn { enumext / #1 }
669 {
670 align .choice:,
671 align / left .code:n =
672 {
673 \tl_clear:c { l__enumext_label_fill_left_#2_tl }
674 \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill }
675 \str_set:cn { l__enumext_align_label_pos_#2_str } { l }
676 },
677 align / right .code:n =
678 {
679 \tl_set:cn { l__enumext_label_fill_left_#2_tl } { \hfill }
680 \tl_clear:c { l__enumext_label_fill_right_#2_tl }
681 \str_set:cn { l__enumext_align_label_pos_#2_str } { r }
682 },
683 align / center .code:n =
684 {
685 \tl_set:cn { l__enumext_label_fill_left_#2_tl } { \hfill }
686 \tl_set:cn { l__enumext_label_fill_right_#2_tl } { \hfill }
687 \str_set:cn { l__enumext_align_label_pos_#2_str } { c }
688 },
689 align / unknown .code:n =
690 \msg_error:nneee { enumext } { unknown-choice }
691 { align } { left, ~ right, ~ center } { \exp_not:n {##1} },
692 align .initial:n = left,
693 align .value_required:n = true,
694 }
695 }
696 \clist_map_inline:nn
697 {
698 {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {keyans}{v}
699 }
700 { __enumext_tmp:nn #1 }
701
702 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
703 {
704 \keys_define:nn { enumext / #1 }
705 {
706 align .choice:,
707 align / left .code:n = \str_set:cn { l__enumext_align_label_#2_str } { l },
708 align / right .code:n = \str_set:cn { l__enumext_align_label_#2_str } { r },
709 align / center .code:n = \str_set:cn { l__enumext_align_label_#2_str } { c },
710 align / unknown .code:n =
711 \msg_error:nneee { enumext } { unknown-choice }
712 { align } { left, ~ right, ~ center } { \exp_not:n {##1} },
713 align .initial:n = left,
714 align .value_required:n = true,
715 }
716 }
717 \clist_map_inline:nn { {enumext*}{vii}, {keyans*}{viii} } { __enumext_tmp:nn #1 }

```

(End of definition for align.)

### 13.14 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 `\label`, the value of the variables `\l__enumext_label_X_tl`, the default values for `\labelwidth` and the “label and ref” system.

### 13.14.1 Define and set label and ref keys for enumext environment

Here we set the default *⟨labels⟩* of the *four levels* of `enumext` environment, along with the default value for `labelwidth` key and `ref` key.

```

label
ref
\l__enumext_label_i_tl 717 \cs_set_protected:Npn __enumext_tmp:nnn #1 #2 #3
\l__enumext_label_ii_tl 718 {
\l__enumext_label_iii_tl 719 \keys_define:nn { enumext / #1 }
\l__enumext_label_iv_tl 720 {
721 label .code:n = {
722 __enumext_label_style:cvn { l__enumext_label_#2_tl }
723 { l__enumext_counter_#2_tl } {##1}
724 \dim_set_eq:cN { l__enumext_labelwidth_#2_dim }
725 \l__enumext_current_widest_dim
726 },
727 label .initial:n = #3,
728 label .value_required:n = true,
729 ref .code:n = __enumext_standar_ref:n {##1},
730 ref .value_required:n = true,
731 }
732 }
733 __enumext_tmp:nnn { level-1 } { i } { \arabic*. }
734 __enumext_tmp:nnn { level-2 } { ii } { (\alph*) }
735 __enumext_tmp:nnn { level-3 } { iii } { \roman*. }
736 __enumext_tmp:nnn { level-4 } { iv } { \Alph*. }

```

(End of definition for `label` and others.)

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.

```

737 \cs_new_protected:Npn __enumext_standar_ref:n #1
738 {
739 \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
740 \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
741 {
742 \msg_error:nnn { enumext } { key-ref-empty } { enumext }
743 }
744 {
745 \tl_set_eq:Nc
746 \l__enumext_ref_the_count_tl { l__enumext_counter_ __enumext_level: _tl }
747 __enumext_regex_counter_style:
748 \tl_set_eq:Nc
749 \l__enumext_ref_the_count_tl { l__enumext_the_counter_ __enumext_level: _tl }
750 \tl_put_right:ce { l__enumext_renew_the_count_ __enumext_level: _tl }
751 {
752 \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl } { \exp_not:V \l__enumext_ref_the_count_tl }
753 }
754 }
755 }

```

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

```

756 \cs_new_protected:Nn __enumext_standar_ref:
757 {
758 \tl_if_empty:cF { l__enumext_renew_the_count_ __enumext_level: _tl }
759 {
760 \tl_use:c { l__enumext_renew_the_count_ __enumext_level: _tl }
761 }
762 }

```

(End of definition for `\__enumext_standar_ref:n` and `\__enumext_standar_ref:`.)

### 13.14.2 Define and set label and ref keys for enumext\* and keyans\* environments

Here we set the default *⟨labels⟩* for `enumext*` and `keyans*` environments, along with the default value for `labelwidth` key and `ref` key.

```

\l__enumext_label_vii_tl 763 \cs_set_protected:Npn __enumext_tmp:nnn #1 #2 #3
\l__enumext_label_viii_tl 764 {
765 \keys_define:nn { enumext / #1 }

```

```

766 {
767 label .code:n = {
768 __enumext_label_style:cvn { l__enumext_label_#2_tl }
769 { l__enumext_counter_#2_tl } {##1}
770 \dim_set_eq:cN { l__enumext_labelwidth_#2_dim }
771 \l__enumext_current_widest_dim
772 },
773 label .initial:n = #3,
774 label .value_required:n = true,
775 ref .code:n = __enumext_starred_ref:n {##1},
776 ref .value_required:n = true,
777 }
778 }
779 __enumext_tmp:nnn { enumext* } { vii } { \arabic*.}
780 __enumext_tmp:nnn { keyans* } { viii } { \Alph*.}

```

(End of definition for `label` and others.)

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

```

__enumext_starred_ref:
781 \cs_new_protected:Npn __enumext_starred_ref:n #1
782 {
783 \tl_set:Nn \l__enumext_ref_key_arg_tl {#1}
784 \int_compare:nNnT { \l__enumext_level_h_int } = { 1 }
785 {
786 \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
787 {
788 \msg_error:nnn { enumext } { key-ref-empty } { enumext* }
789 }
790 {
791 \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_vii_tl
792 __enumext_regex_counter_style:
793 \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_the_counter_vii_tl
794 \tl_put_right:Ne \l__enumext_renew_the_count_vii_tl
795 {
796 \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl } { \exp_not:V
797 }
798 }
799 }
800 \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
801 {
802 \tl_if_empty:NTF \l__enumext_ref_key_arg_tl
803 {
804 \msg_error:nnn { enumext } { key-ref-empty } { keyans* }
805 }
806 {
807 \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_counter_viii_tl
808 __enumext_regex_counter_style:
809 \tl_set_eq:NN \l__enumext_ref_the_count_tl \l__enumext_the_counter_viii_tl
810 \tl_put_right:Ne \l__enumext_renew_the_count_viii_tl
811 {
812 \exp_not:N \renewcommand { \exp_not:V \l__enumext_ref_the_count_tl } { \exp_not:V
813 }
814 }
815 }
816 }

```

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.

```

817 \cs_new_protected:Npn __enumext_starred_ref:
818 {
819 \int_compare:nNnT { \l__enumext_level_h_int } = { 1 }
820 {
821 \tl_if_empty:NF \l__enumext_renew_the_count_vii_tl
822 {
823 \tl_use:N \l__enumext_renew_the_count_vii_tl
824 }
825 }
826 \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
827 {
828 \tl_if_empty:NF \l__enumext_renew_the_count_viii_tl
829 {
830 \tl_use:N \l__enumext_renew_the_count_viii_tl

```

```

831 }
832 }
833 }

```

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

### 13.14.3 Define and set label and ref keys for keyans and keyanspic environments

Here we set the default *label* for `keyans` and `keyanspic` environment, along with the default value for `labelwidth` and `ref` key. The `keyanspic` environment use the same *label* as the `keyans` environment.

```

__enumext_label_v_tl 834 \keys_define:nn { enumext / keyans }
__enumext_label_vi_tl 835 {
836 label .code:n = {
837 __enumext_label_style:cvn { __enumext_label_v_tl }
838 { __enumext_counter_v_tl } {#1}
839 \dim_set_eq:cN { __enumext_labelwidth_v_dim }
840 __enumext_current_widest_dim
841 __enumext_label_style:cvn { __enumext_label_vi_tl }
842 { __enumext_counter_vi_tl } {#1}
843 \dim_set_eq:cN { __enumext_labelwidth_v_dim }
844 __enumext_current_widest_dim
845 },
846 label .initial:n = \Alph*,
847 label .value_required:n = true,
848 ref .code:n = __enumext_keyans_ref:n {#1},
849 ref .value_required:n = true,
850 }

```

(End of definition for `label` and others.)

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

```

__enumext_keyans_ref:n 851 \cs_new_protected:Npn __enumext_keyans_ref:n #1
__enumext_keyans_ref: 852 {
853 \tl_set:Nn __enumext_ref_key_arg_tl {#1}
854 \tl_if_empty:NTF __enumext_ref_key_arg_tl
855 {
856 \msg_error:nnn { enumext } { key-ref-empty } { keyans }
857 }
858 {
859 \tl_set_eq:NN __enumext_ref_the_count_tl __enumext_counter_v_tl
860 __enumext_regex_counter_style:
861 \tl_set_eq:NN __enumext_ref_the_count_tl __enumext_the_counter_v_tl
862 \tl_put_right:Ne __enumext_renew_the_count_v_tl
863 {
864 \exp_not:N \renewcommand { \exp_not:V __enumext_ref_the_count_tl } { \exp_not:V \l_
865 }
866 }
867 }

```

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

```

868 \cs_new_protected:Nn __enumext_keyans_ref:
869 {
870 \tl_if_empty:NF __enumext_renew_the_count_v_tl
871 {
872 \tl_use:N __enumext_renew_the_count_v_tl
873 }
874 }

```

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

### 13.15 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: __enumext_label_X_tl
#2: __enumext_start_X_int
#3: <integer or string>

```

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 *integer* or *string* of the form `\Alph`, `\alph`, `\Roman` or `\roman`. This effectively allows `start=A` or `start=1` to be used.

```

875 \cs_new_protected:Npn __enumext_start_from:NNn #1 #2 #3
876 {

```

```

877 __enumext_if_is_int:nTF { #3 }
878 {
879 \int_set:Nn #2 {#3}
880 }
881 {
882 \regex_match:nVT { \c{Alpha} | \c{alpha} } {#1}
883 { \int_set:Nn #2 { \int_from_alph:n {#3} } }
884 \regex_match:nVT { \c{Roman} | \c{roman} } {#1}
885 { \int_set:Nn #2 { \int_from_roman:n {#3} } }
886 }
887 }
888 \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

```

The function \\_\_enumext\_widest\_from:nNNn used by the `widest` key take four arguments:

- #1: The counter associated with the environment level
- #2: \\_\_enumext\_label\_X\_tl
- #3: \\_\_enumext\_labelwidth\_X\_dim
- #4: *integer or string*

The second and third arguments of this function are the values set by `label` and `labelwidth` keys, the four argument can be an *integer* or *string* of the form `\Alpha`, `\alpha`, `\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.

```

889 \cs_new_protected:Npn __enumext_widest_from:nNNn #1 #2 #3 #4
890 {
891 __enumext_if_is_int:nTF {#4}
892 {
893 \setcounter{enumX#1} { #4 }
894 }
895 {
896 \regex_match:nVT { \c{Alpha} | \c{alpha} } {#2}
897 { \setcounter{enumX#1} { \int_from_alph:n {#4} } }
898 \regex_match:nVT { \c{Roman} | \c{roman} } {#2}
899 { \setcounter{enumX#1} { \int_from_roman:n {#4} } }
900 }
901 __enumext_label_width_by_box:cv
902 { __enumext_labelwidth_#1_dim } { __enumext_label_#1_tl }
903 }
904 \cs_generate_variant:Nn __enumext_widest_from:nNNn { nccn }

```

(End of definition for \\_\_enumext\_widest\_from:nNNn.)

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

```

start
start*
widest
905 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
906 {
907 \keys_define:nn { enumext / #1 }
908 {
909 start* .code:n = {
910 __enumext_start_from:ccn
911 { __enumext_label_#2_tl }
912 { __enumext_start_#2_int } {##1}
913 },
914 start* .value_required:n = true,
915 start .code:n = {
916 __enumext_start_from:cce
917 { __enumext_label_#2_tl }
918 { __enumext_start_#2_int } { \int_eval:n {##1} }
919 },
920 start .initial:n = 1,
921 start .value_required:n = true,
922 widest .code:n = {
923 __enumext_widest_from:nccn {#2}
924 { __enumext_label_#2_tl }
925 { __enumext_labelwidth_#2_dim } {##1}
926 },
927 widest .value_required:n = true,
928 }
929 }
930 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

```



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

### 13.16 Setting keys for vertical spaces

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

```

topsep
partopsep
parsep
noitemsep
nosep
931 \cs_set_protected:Npn __enumext_tmp:nnnnnn #1 #2 #3 #4 #5 #6
932 {
933 \keys_define:nn { enumext / #1 }
934 {
935 topsep .skip_set:c = { l__enumext_topsep_#2_skip },
936 topsep .initial:n = {#3},
937 topsep .value_required:n = true,
938 partopsep .skip_set:c = { l__enumext_partopsep_#2_skip },
939 partopsep .initial:n = {#4},
940 partopsep .value_required:n = true,
941 parsep .skip_set:c = { l__enumext_parsep_#2_skip },
942 parsep .initial:n = {#5},
943 parsep .value_required:n = true,
944 itemsep .skip_set:c = { l__enumext_itemsep_#2_skip },
945 itemsep .initial:n = {#6},
946 itemsep .value_required:n = true,
947 noitemsep .meta:n = { itemsep = 0pt, parsep = 0pt },
948 noitemsep .value_forbidden:n = true,
949 nosep .meta:n = {
950 itemsep = 0pt, parsep = 0pt,
951 topsep = 0pt, partopsep = 0pt,
952 },
953 nosep .value_forbidden:n = true,
954 }
955 }
```

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

```

956 __enumext_tmp:nnnnnn { level-1 } { i } { 8.0pt plus 2.0pt minus 4.0pt }
957 { 2.0pt plus 1.0pt minus 1.0pt } { 4.0pt plus 2.0pt minus 1.0pt }
958 { 4.0pt plus 2.0pt minus 1.0pt }
959 __enumext_tmp:nnnnnn { level-2 } { ii } { 4.0pt plus 2.0pt minus 1.0pt }
960 { 2.0pt plus 1.0pt minus 1.0pt } { 2.0pt plus 1.0pt minus 1.0pt }
961 { 2.0pt plus 1.0pt minus 1.0pt }
962 __enumext_tmp:nnnnnn { level-3 } { iii } { 2.0pt plus 1.0pt minus 1.0pt }
963 { 1.0pt minus 1.0pt } { 0pt } { 2.0pt plus 1.0pt minus 1.0pt }
964 __enumext_tmp:nnnnnn { level-4 } { iv } { 2.0pt plus 1.0pt minus 1.0pt }
965 { 1.0pt minus 1.0pt } { 0pt } { 2.0pt plus 1.0pt minus 1.0pt }
966 __enumext_tmp:nnnnnn { keyans } { v } { 4.0pt plus 2.0pt minus 1.0pt }
967 { 2.0pt plus 1.0pt minus 1.0pt } { 2.0pt plus 1.0pt minus 1.0pt }
968 { 2.0pt plus 1.0pt minus 1.0pt }
969 __enumext_tmp:nnnnnn { enumext* } { vii } { 8.0pt plus 2.0pt minus 4.0pt }
970 { 2.0pt plus 1.0pt minus 1.0pt } { 4.0pt plus 2.0pt minus 1.0pt }
971 { 4.0pt plus 2.0pt minus 1.0pt }
972 __enumext_tmp:nnnnnn { keyans* } { viii } { 4.0pt plus 2.0pt minus 1.0pt }
973 { 2.0pt plus 1.0pt minus 1.0pt } { 2.0pt plus 1.0pt minus 1.0pt }
974 { 2.0pt plus 1.0pt minus 1.0pt }
```

(End of definition for `topsep` and others.)

### 13.17 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:` apply `\vspace[-\baselineskip]` and set `\topsep=0pt` for the “first level” of the nested `enumext` environment.

base-fix

We define the key `base-fix` only for the “first level” of `enumext` environment.

```

__enumext_nested_base_line_fix:
975 \keys_define:nn { enumext / level-1 }
976 {
977 base-fix .bool_set:N = \l__enumext_base_line_fix_bool,
978 base-fix .initial:n = false,
979 base-fix .value_forbidden:n = true,
980 }
```

The function `\__enumext_nested_base_line_fix:` passed to the `\__enumext_parse_keys:n` function in the definition of the `enumext` environment (§13.39) will be responsible for applying the *baseline correction* and adjusting the *⟨keys⟩* for the `enumext` environment and the `\printkeyans` with *starred argument* ‘\*’ (§13.47).

We will first implement the function code from the user side of the `base-fix` key, that is, only the user knows when it is necessary to apply it within the document in which case the variable `\__enumext_print_keyans_star_bool` set by the `\printkeyans` command is false and the variable `\__enumext_base_line_fix_bool` is true.

```

981 \cs_new_protected:Nn __enumext_nested_base_line_fix:
982 {
983 \bool_lazy_all:nT
984 {
985 { \bool_if_p:N __enumext_starred_first_bool }
986 { \bool_if_p:N __enumext_base_line_fix_bool }
987 { \bool_not_p:n { __enumext_print_keyans_star_bool } }
988 }
989 {
990 \mode_leave_vertical:
991 \vspace { -\dim_eval:n { \baselineskip + \parsep } }
992 }

```

When we are running the `\printkeyans` command with the *starred argument* ‘\*’ the variable `\__enumext_print_keyans_star_bool` is true and we can run a simplified version of `\vspace` using `\skip_vertical:n`.

```

993 \bool_lazy_and:nnT
994 { \bool_if_p:N __enumext_starred_first_bool }
995 { \bool_if_p:N __enumext_print_keyans_star_bool }
996 {
997 \mode_leave_vertical:
998 \skip_vertical:n { -\baselineskip }
999 \skip_vertical:N \c_zero_skip
1000 }

```

Finally we set the values of the keys `topsep`, `above` and `above*` for the “first level” of `enumext` environment equal to `0pt` and set the variable `\__enumext_base_line_fix_bool` to false.

```

1001 \keys_set:nn { enumext / level-1 }
1002 {
1003 topsep = 0pt, above = 0pt, above* = 0pt,
1004 }
1005 \bool_set_false:N __enumext_base_line_fix_bool
1006 }

```

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

### 13.18 Setting keys for horizontal spaces

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

```

1007 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1008 {
1009 \keys_define:nn { enumext / #1 }
1010 {
1011 itemindent .dim_set:c = { __enumext_fake_item_indent_#2_dim },
1012 itemindent .value_required:n = true,
1013 rightmargin .dim_set:c = { __enumext_rightmargin_#2_dim },
1014 rightmargin .value_required:n = true,
1015 listparindent .dim_set:c = { __enumext_listparindent_#2_dim },
1016 listparindent .value_required:n = true,
1017 list-offset .dim_set:c = { __enumext_listoffset_#2_dim },
1018 list-offset .value_required:n = true,
1019 list-indent .code:n =
1020 \bool_set_true:c { __enumext_leftmargin_tmp_#2_bool }
1021 \dim_set:cn { __enumext_leftmargin_tmp_#2_dim } {##1},
1022 list-indent .value_required:n = true,
1023 }
1024 }
1025 \clist_map_inline:nn
1026 {
1027 {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {keyans}{v}
1028 }
1029 { __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.

```

1030 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1031 {
1032 \keys_define:nn { enumext / #1 }
1033 {
1034 itemindent .dim_set:c = { l__enumext_fake_item_indent_#2_dim },
1035 itemindent .value_required:n = true,
1036 rightmargin .dim_set:c = { l__enumext_rightmargin_#2_dim },
1037 rightmargin .value_required:n = true,
1038 listparindent .dim_set:c = { l__enumext_listparindent_#2_dim },
1039 listparindent .value_required:n = true,
1040 list-offset .dim_set:c = { l__enumext_listoffset_#2_dim },
1041 list-offset .value_required:n = true,
1042 list-indent .meta:n = { list-offset = ##1 },
1043 list-indent .value_required:n = true,
1044 }
1045 }
1046 \clist_map_inline:nn
1047 {
1048 {enumext*}{vii}, {keyans*}{viii}
1049 }
1050 { __enumext_tmp:nn #1 }

```

### 13.18.1 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 `0pt`. 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.

```

1051 \cs_set_protected:Nn __enumext_fake_item_indent:
1052 {
1053 \dim_compare:nNnT
1054 { \dim_use:c { l__enumext_fake_item_indent_ __enumext_level: _dim } }
1055 >
1056 { \c_zero_dim }
1057 {
1058 \tl_set:ce { l__enumext_fake_item_indent_ __enumext_level: _tl }
1059 {
1060 \exp_not:N \mode_leave_vertical:
1061 \exp_not:n { \skip_horizontal:n }
1062 { \dim_use:c { l__enumext_fake_item_indent_ __enumext_level: _dim } }
1063 \exp_not:N \ignorespaces
1064 }
1065 }
1066 }
1067 \cs_set_protected:Nn __enumext_keyans_fake_item_indent:
1068 {
1069 \dim_compare:nNnT
1070 { \l__enumext_fake_item_indent_v_dim } > { \c_zero_dim }
1071 {
1072 \tl_set:Nc \l__enumext_fake_item_indent_v_tl
1073 {
1074 \exp_not:N \mode_leave_vertical:
1075 \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_v_dim
1076 \exp_not:N \ignorespaces
1077 }
1078 }
1079 }
1080 \cs_set_protected:Nn __enumext_fake_item_indent_vii:
1081 {
1082 \dim_compare:nNnT
1083 { \l__enumext_fake_item_indent_vii_dim } > { \c_zero_dim }
1084 {
1085 \tl_set:Nc \l__enumext_fake_item_indent_vii_tl
1086 {
1087 \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_vii_dim
1088 \exp_not:N \ignorespaces
1089 }
1090 }

```

```

1091 }
1092 \cs_set_protected:Nn __enumext_fake_item_indent_viii:
1093 {
1094 \dim_compare:nNt
1095 { \l__enumext_fake_item_indent_viii_dim } > { \c_zero_dim }
1096 {
1097 \tl_set:Nc \l__enumext_fake_item_indent_viii_tl
1098 {
1099 \exp_not:N \skip_horizontal:N \l__enumext_fake_item_indent_viii_dim
1100 \exp_not:N \ignorespaces
1101 }
1102 }
1103 }

```

(End of definition for `\__enumext_fake_item_indent:` and others.)

### 13.19 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.

```

1104 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1105 {
1106 \keys_define:nn { enumext / #1 }
1107 {
1108 show-length .bool_set:c = { \l__enumext_show_length_#2_bool },
1109 show-length .initial:n = false,
1110 }
1111 }
1112 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

```

(End of definition for `show-length`.)

### 13.20 Setting before, after and first keys

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

```

1113 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1114 {
1115 \keys_define:nn { enumext / #1 }
1116 {
1117 before .tl_set:c = { \l__enumext_before_no_starred_key_#2_tl },
1118 before .value_required:n = true,
1119 before* .tl_set:c = { \l__enumext_before_starred_key_#2_tl },
1120 before* .value_required:n = true,
1121 after .tl_set:c = { \l__enumext_after_stop_list_#2_tl },
1122 after .value_required:n = true,
1123 first .tl_set:c = { \l__enumext_after_list_args_#2_tl },
1124 first .value_required:n = true,
1125 }
1126 }
1127 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

```

(End of definition for `before` and others.)

#### 13.20.1 Functions for before, after and first keys in enumext

`\__enumext_before_args_exec:` The function `\__enumext_before_args_exec:` executes the `{\code}` set by the `before*` key “before” the `enumext` environment is started. The `{\code}` is executed “without” knowing any definition of the `{\arg two}` of the list: `{\code}\list{\arg one}{\arg two}`.

```

1128 \cs_new_protected:Nn __enumext_before_args_exec:
1129 {
1130 \tl_use:c { \l__enumext_before_starred_key_ __enumext_level: _tl }
1131 }

```

The function `\__enumext_before_keys_exec:` executes the `{\code}` set by the `before` key “before” the `enumext` environment is started in *second argument* of the list. The `{\code}` is executed “knowing” all definition and values provides by `\keys: \list{\arg one}{\arg two}{\code}`

```

1132 \cs_new_protected:Nn __enumext_before_keys_exec:
1133 {
1134 \tl_use:c { \l__enumext_before_no_starred_key_ __enumext_level: _tl }
1135 }

```

The function `\__enumext_after_stop_list:` executes the `{\code}` set by the `after` key “after” the `enumext` environment has finished: `\endlist{\code}`.

```
1136 \cs_new_protected:Nn __enumext_after_stop_list:
1137 {
1138 \tl_use:c { l__enumext_after_stop_list_ __enumext_level: _tl }
1139 }
```

The function `\__enumext_after_args_exec:` executes the `{\code}` 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}{\code}\item`.

```
1140 \cs_new_protected:Nn __enumext_after_args_exec:
1141 {
1142 \tl_use:c { l__enumext_after_list_args_ __enumext_level: _tl }
1143 }
```

(End of definition for `\__enumext_before_args_exec:` and others.)

### 13.20.2 Functions for before, after and first keys in keyans

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

```
__enumext_before_args_exec_v:
__enumext_before_keys_exec_v:
__enumext_after_stop_list_v:
__enumext_after_args_exec_v:
1144 \cs_new_protected:Nn __enumext_before_args_exec_v:
1145 {
1146 \tl_use:N \l__enumext_before_starred_key_v_tl
1147 }
1148 \cs_new_protected:Nn __enumext_before_keys_exec_v:
1149 {
1150 \tl_use:N \l__enumext_before_no_starred_key_v_tl
1151 }
1152 \cs_new_protected:Nn __enumext_after_stop_list_v:
1153 {
1154 \tl_use:N \l__enumext_after_stop_list_v_tl
1155 }
1156 \cs_new_protected:Nn __enumext_after_args_exec_v:
1157 {
1158 \tl_use:N \l__enumext_after_list_args_v_tl
1159 }
```

(End of definition for `\__enumext_before_args_exec_v:` and others.)

### 13.20.3 Functions for before, after and first keys in enumext\* and keyans\*

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

```
__enumext_before_args_exec_vii:
__enumext_before_keys_exec_vii
__enumext_after_stop_list_vii:
__enumext_after_args_exec_vii:
1160 \cs_new_protected:Nn __enumext_before_args_exec_vii:
1161 {
1162 \tl_use:N \l__enumext_before_starred_key_vii_tl
1163 }
1164 \cs_new_protected:Nn __enumext_before_args_exec_viii:
1165 {
1166 \tl_use:N \l__enumext_before_starred_key_viii_tl
1167 }
1168 \cs_new_protected:Nn __enumext_before_keys_exec_vii:
1169 {
1170 \tl_use:N \l__enumext_before_no_starred_key_vii_tl
1171 }
1172 \cs_new_protected:Nn __enumext_before_keys_exec_viii:
1173 {
1174 \tl_use:N \l__enumext_before_no_starred_key_viii_tl
1175 }
1176 \cs_new_protected:Nn __enumext_after_stop_list_vii:
1177 {
1178 \tl_use:N \l__enumext_after_stop_list_vii_tl
1179 }
1180 \cs_new_protected:Nn __enumext_after_stop_list_viii:
1181 {
1182 \tl_use:N \l__enumext_after_stop_list_viii_tl
1183 }
1184 \cs_new_protected:Nn __enumext_after_args_exec_vii:
1185 {
1186 \tl_use:N \l__enumext_after_list_args_vii_tl
1187 }
1188 \cs_new_protected:Nn __enumext_after_args_exec_viii:
1189 {
1190 \tl_use:N \l__enumext_after_list_args_viii_tl
1191 }
```

(End of definition for `\__enumext_before_args_exec_vii`: and others.)

### 13.21 Setting keys for multicol and minipage

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.

```

1192 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1193 {
1194 \keys_define:nn { enumext / #1 }
1195 {
1196 mini-env .dim_set:c = { __enumext_minipage_right_#2_dim },
1197 mini-env .value_required:n = true,
1198 mini-sep .dim_set:c = { __enumext_minipage_hsep_#2_dim },
1199 mini-sep .initial:n = 0.3333em,
1200 mini-sep .value_required:n = true,
1201 columns-sep .dim_set:c = { __enumext_columns_sep_#2_dim },
1202 columns-sep .value_required:n = true,
1203 columns .int_set:c = { __enumext_columns_#2_int },
1204 columns .initial:n = 1,
1205 columns .value_required:n = true,
1206 }
1207 }
1208 \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.

```

1209 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
1210 {
1211 \keys_define:nn { enumext / #1 }
1212 {
1213 mini-right .tl_gset:c = { g__enumext_miniright_code_#2_tl },
1214 mini-right .value_required:n = true,
1215 mini-right* .code:n = {
1216 \bool_gset_true:c { g__enumext_minipage_center_#2_bool }
1217 \keys_set:nn { enumext / #1 } { mini-right = {##1} }
1218 },
1219 mini-right* .value_required:n = true,
1220 }
1221 }
1222 \clist_map_inline:nn { {enumext*}{vii}, {keyans*}{viii} } { __enumext_tmp:nn #1 }

```

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

### 13.22 Adjustment of vertical spaces for multicol

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



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

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

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



### 13.22.1 Adjustment of vertical spaces for multicol 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 `multicol` environment in `enumext`.

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

Set the values of `\l__enumext_multicols_above_X_skip` and `\l__enumext_multicols_below_X_skip` equal to the value of `\topsep` in the *current level*.

```

1223 \cs_new_protected:Nn __enumext_multi_set_vskip:
1224 {
1225 \skip_set:cn { \l__enumext_multicols_above_ __enumext_level: _skip }
1226 {
1227 \skip_use:c { \l__enumext_topsep_ __enumext_level: _skip }
1228 }
1229 \skip_set:cn { \l__enumext_multicols_below_ __enumext_level: _skip }
1230 {
1231 \skip_use:c { \l__enumext_topsep_ __enumext_level: _skip }
1232 }
1233 __enumext_add_pre_parsep:
1234 }

```

(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*.

```

1235 \cs_new_protected:Nn __enumext_add_pre_parsep:
1236 {
1237 \int_case:nn { \l__enumext_level_int }
1238 {
1239 { 2 }{
1240 \skip_if_eq:nnF { \l__enumext_parsep_i_skip } { \c_zero_skip }
1241 {
1242 \skip_add:Nn \l__enumext_multicols_above_ii_skip
1243 {
1244 \l__enumext_parsep_i_skip
1245 }
1246 }
1247 }
1248 { 3 }{
1249 \skip_if_eq:nnF { \l__enumext_parsep_ii_skip } { \c_zero_skip }
1250 {
1251 \skip_add:Nn \l__enumext_multicols_above_iii_skip
1252 {
1253 \l__enumext_parsep_ii_skip
1254 }
1255 }
1256 }
1257 { 4 }{
1258 \skip_if_eq:nnF { \l__enumext_parsep_iii_skip } { \c_zero_skip }
1259 {
1260 \skip_add:Nn \l__enumext_multicols_above_iv_skip
1261 {
1262 \l__enumext_parsep_iii_skip
1263 }
1264 }
1265 }
1266 }
1267 }

```

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

`\__enumext_multi_addvspace:` The function `\__enumext_multi_addvspace:` will apply the spaces set using `\addvspace` “*above*” the `multicol` environment in `enumext`, taking into account whether  $\text{\TeX}$  is in  $\langle \textit{horizontal mode} \rangle$  or  $\langle \textit{vertical mode} \rangle$ .

```

1268 \cs_new_protected:Nn __enumext_multi_addvspace:
1269 {
1270 __enumext_multi_set_vskip:
1271 \mode_if_vertical:T
1272 {

```

```

1273 \skip_add:cn { __enumext_multicols_above_ __enumext_level: _skip }
1274 {
1275 \skip_use:c { __enumext_partopsep_ __enumext_level: _skip }
1276 }
1277 \skip_add:cn { __enumext_multicols_below_ __enumext_level: _skip }
1278 {
1279 \skip_use:c { __enumext_partopsep_ __enumext_level: _skip }
1280 }
1281 }
1282 \par\nopagebreak
1283 \addvspace{ \skip_use:c { __enumext_multicols_above_ __enumext_level: _skip } }
1284 }

```

(End of definition for `\__enumext_multi_addvspace:`)

### 13.22.2 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`.

```

1285 \cs_new_protected:Nn __enumext_keyans_multi_set_vskip:
1286 {
1287 \skip_set:Nn __enumext_multicols_above_v_skip
1288 {
1289 __enumext_topsep_v_skip
1290 }
1291 \skip_set:Nn __enumext_multicols_below_v_skip
1292 {
1293 __enumext_topsep_v_skip
1294 }
1295 }
1296 \cs_new_protected:Nn __enumext_keyans_multi_addvspace:
1297 {
1298 __enumext_keyans_multi_set_vskip:
1299 \mode_if_vertical:T
1300 {
1301 \skip_add:Nn __enumext_multicols_above_v_skip
1302 {
1303 \skip_use:N __enumext_partopsep_v_skip
1304 }
1305 \skip_add:Nn __enumext_multicols_below_v_skip
1306 {
1307 \skip_use:N __enumext_partopsep_v_skip
1308 }
1309 }
1310 \par\nopagebreak
1311 \addvspace{ __enumext_multicols_above_v_skip }
1312 }

```

(End of definition for `\__enumext_keyans_multi_set_vskip:` and `\__enumext_keyans_multi_addvspace:`)

### 13.23 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.

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=0pt` and to this is added the `\partopsep` parameter that comes into action according to whether  $\TeX$  is in *horizontal mode* or *vertical mode*. 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.

### 13.23.1 Adjustment of vertical spaces for minipage in enumext

`\__enumext_minipage_set_skip:`  
`\__enumext_minipage_add_space:`

The function `\__enumext_minipage_set_skip:` will take care of determining the “*adjust*” spaces that we will apply “*above*” and “*below*” the `__enumext_mini_page` environment in `enumext`.

First we will set the value of `\l__enumext_minipage_right_skip` equal to `\topsep`, then we will see if  $\text{\TeX}$  is in *vertical mode* and we will add `\partopsep`, followed by that we set the value of `\l__enumext_minipage_after_skip`.

```

1313 \cs_new_protected:Nn __enumext_minipage_set_skip:
1314 {
1315 \skip_set:Nn \l__enumext_minipage_right_skip
1316 {
1317 \skip_use:c { \l__enumext_topsep_ __enumext_level: _skip }
1318 }
1319 \mode_if_vertical:T
1320 {
1321 \skip_add:Nn \l__enumext_minipage_right_skip
1322 {
1323 \skip_use:c { \l__enumext_partopsep_ __enumext_level: _skip }
1324 }
1325 }
1326 \skip_set_eq:NN \l__enumext_minipage_after_skip \l__enumext_minipage_right_skip

```

We will adjust the values `\l__enumext_multicols_above_X_skip` and `\l__enumext_multicols_below_X_skip` and call the function `\__enumext_pre_itemsep_skip:`.

```

1327 \skip_set_eq:cN
1328 { \l__enumext_multicols_above_ __enumext_level: _skip } \l__enumext_minipage_right_skip
1329 \skip_set_eq:cN
1330 { \l__enumext_multicols_below_ __enumext_level: _skip } \l__enumext_minipage_right_skip
1331 __enumext_pre_itemsep_skip:

```

If the environment `multicols` is active, we set `\topskip=0pt` and then we make `\multicolsep` have the same value as `\l__enumext_multicols_above_X_skip`.

```

1332 \int_compare:nNnT
1333 { \int_use:c { \l__enumext_columns_ __enumext_level: _int } } > { 1 }
1334 {
1335 \skip_zero:N \topskip
1336 \skip_set_eq:Nc \multicolsep { \l__enumext_multicols_above_ __enumext_level: _skip }
1337 }
1338 }

```

The function `\__enumext_minipage_add_space:` will apply the spaces on the “*left side*” using `\addvspace` “*above*” the `__enumext_mini_page` environment, taking into account whether  $\text{\TeX}$  is in *horizontal mode* or *vertical mode*. Here we use the plain  $\text{\TeX}$  macro `\nointerlineskip` to prevent baseline “*glue*” being added between the next pair of boxes in a *vertical list*. For the latter we will make some adjustments since the `\partopsep` parameter comes into play and this affects the *vertical spacing*.

```

1339 \cs_new_protected:Nn __enumext_minipage_add_space:
1340 {
1341 __enumext_minipage_set_skip:
1342 __enumext_unskip_unkern:
1343 \mode_if_vertical:TF
1344 {
1345 \nopagebreak\nointerlineskip
1346 }
1347 {
1348 \par\nopagebreak\nointerlineskip
1349 \skip_zero:c { \l__enumext_partopsep_ __enumext_level: _skip }
1350 }
1351 \int_compare:nNnTF
1352 { \int_use:c { \l__enumext_columns_ __enumext_level: _int } } > { 1 }
1353 {
1354 \addvspace{ 0.445\box_ht:N \strutbox }
1355 }
1356 {
1357 \addvspace{ 0.250\box_ht:N \strutbox }
1358 }
1359 }

```

(End of definition for `\__enumext_minipage_set_skip:` and `\__enumext_minipage_add_space:.`)

`\__enumext_pre_itemsep_skip:` The function `\__enumext_pre_itemsep_skip:` will adjust the spaces below the environment `minipage` and the environment `multicols` if it is nested in it, taking into account the value of `\itemsep` from the previous level.

```

1360 \cs_new_protected:Nn __enumext_pre_itemsep_skip:
1361 {
1362 \int_case:nn { \l__enumext_level_int }
1363 {
1364 { 2 }{
1365 \skip_if_eq:nnTF
1366 { \l__enumext_itemsep_i_skip } { \l__enumext_minipage_after_skip }
1367 {
1368 \skip_set:Nn \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1369 \skip_set:Nn \l__enumext_multicols_below_ii_skip { 0.350\box_ht:N \strutbox }
1370 }
1371 {
1372 \dim_compare:nNnT
1373 { \l__enumext_itemsep_i_skip } < { \l__enumext_minipage_after_skip }
1374 {
1375 \skip_sub:Nn
1376 \l__enumext_minipage_after_skip { \l__enumext_itemsep_i_skip }
1377 \skip_sub:Nn
1378 \l__enumext_multicols_below_ii_skip { \l__enumext_itemsep_i_skip }
1379 \skip_add:Nn
1380 \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1381 \skip_add:Nn
1382 \l__enumext_multicols_below_ii_skip { 0.350\box_ht:N \strutbox }
1383 }
1384 \dim_compare:nNnT
1385 { \l__enumext_itemsep_i_skip } > { \l__enumext_minipage_after_skip }
1386 {
1387 \skip_set:Nn \l__enumext_minipage_temp_skip
1388 {
1389 \l__enumext_itemsep_i_skip - \l__enumext_minipage_after_skip
1390 }
1391 \skip_sub:Nn
1392 \l__enumext_minipage_after_skip { \l__enumext_itemsep_i_skip }
1393 \skip_sub:Nn
1394 \l__enumext_multicols_below_ii_skip { \l__enumext_itemsep_i_skip }
1395 \skip_add:Nn
1396 \l__enumext_minipage_after_skip
1397 { 0.150\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1398 \skip_add:Nn
1399 \l__enumext_multicols_below_ii_skip
1400 { 0.350\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1401 }
1402 }
1403 }
1404 { 3 }{
1405 \skip_if_eq:nnTF
1406 { \l__enumext_itemsep_ii_skip } { \c_zero_skip }
1407 {
1408 \skip_set:Nn \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1409 \skip_set:Nn \l__enumext_multicols_below_iii_skip { 0.350\box_ht:N \strutbox }
1410 }
1411 {
1412 \dim_compare:nNnT
1413 { \l__enumext_itemsep_ii_skip } < { \l__enumext_minipage_after_skip }
1414 {
1415 \skip_sub:Nn
1416 \l__enumext_minipage_after_skip { \l__enumext_itemsep_ii_skip }
1417 \skip_sub:Nn
1418 \l__enumext_multicols_below_iii_skip { \l__enumext_itemsep_ii_skip }
1419 \skip_add:Nn
1420 \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1421 \skip_add:Nn
1422 \l__enumext_multicols_below_iii_skip { 0.350\box_ht:N \strutbox }
1423 }
1424 \dim_compare:nNnT
1425 { \l__enumext_itemsep_ii_skip } > { \l__enumext_minipage_after_skip }

```

```

1426 {
1427 \skip_set:Nn \l__enumext_minipage_temp_skip
1428 {
1429 \l__enumext_itemsep_ii_skip - \l__enumext_minipage_after_skip
1430 }
1431 \skip_sub:Nn
1432 \l__enumext_minipage_after_skip { \l__enumext_itemsep_ii_skip }
1433 \skip_sub:Nn
1434 \l__enumext_multicols_below_iii_skip { \l__enumext_itemsep_ii_skip }
1435 \skip_add:Nn
1436 \l__enumext_minipage_after_skip
1437 { 0.150\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1438 \skip_add:Nn
1439 \l__enumext_multicols_below_iii_skip
1440 { 0.350\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1441 }
1442 }
1443 }
1444 { 4 }{
1445 \skip_if_eq:nnTF { \l__enumext_itemsep_iii_skip } { \c_zero_skip }
1446 {
1447 \skip_set:Nn \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1448 \skip_set:Nn \l__enumext_multicols_below_iv_skip { 0.350\box_ht:N \strutbox }
1449 }
1450 {
1451 \dim_compare:nNnT
1452 { \l__enumext_itemsep_iii_skip } < { \l__enumext_minipage_after_skip }
1453 {
1454 \skip_sub:Nn
1455 \l__enumext_minipage_after_skip { \l__enumext_itemsep_iii_skip }
1456 \skip_sub:Nn
1457 \l__enumext_multicols_below_iv_skip { \l__enumext_itemsep_iii_skip }
1458 \skip_add:Nn
1459 \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1460 \skip_add:Nn
1461 \l__enumext_multicols_below_iv_skip { 0.350\box_ht:N \strutbox }
1462 }
1463 \dim_compare:nNnT
1464 { \l__enumext_itemsep_iii_skip } > { \l__enumext_minipage_after_skip }
1465 {
1466 \skip_set:Nn \l__enumext_minipage_temp_skip
1467 {
1468 \l__enumext_itemsep_iii_skip - \l__enumext_minipage_after_skip
1469 }
1470 \skip_sub:Nn
1471 \l__enumext_minipage_after_skip { \l__enumext_itemsep_iii_skip }
1472 \skip_sub:Nn
1473 \l__enumext_multicols_below_iv_skip { \l__enumext_itemsep_iii_skip }
1474 \skip_add:Nn
1475 \l__enumext_minipage_after_skip
1476 { 0.150\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1477 \skip_add:Nn
1478 \l__enumext_multicols_below_iv_skip
1479 { 0.350\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1480 }
1481 }
1482 }
1483 }
1484 }

```

(End of definition for `\__enumext_pre_itemsep_skip:`)

### 13.23.2 Adjustment of vertical spaces for minipage in keyans

```

__enumext_keyans_minipage_set_skip:
__enumext_keyans_minipage_add_space:
__enumext_keyans_pre_itemsep_skip:

```

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_page` environment in [keyans](#). The implementation of this function is the same as the one used in [enumext](#).

```

1485 \cs_new_protected:Nn __enumext_keyans_minipage_set_skip:
1486 {
1487 \skip_zero:N \l__enumext_minipage_after_skip
1488 \skip_zero:N \l__enumext_minipage_left_skip
1489 \skip_zero:N \l__enumext_minipage_right_skip

```

```

1490 \skip_set:Nn \l__enumext_minipage_right_skip
1491 {
1492 \l__enumext_topsep_v_skip
1493 }
1494 \mode_if_vertical:T
1495 {
1496 \skip_add:Nn \l__enumext_minipage_right_skip
1497 {
1498 \l__enumext_partopsep_v_skip
1499 }
1500 }
1501 \skip_set_eq:NN \l__enumext_minipage_after_skip \l__enumext_minipage_right_skip
1502 \skip_set_eq:NN \l__enumext_multicols_above_v_skip \l__enumext_minipage_right_skip
1503 \skip_set_eq:NN \l__enumext_multicols_below_v_skip \l__enumext_minipage_right_skip
1504 __enumext_keyans_pre_itemsep_skip:
1505 \int_compare:nNnT { \l__enumext_columns_v_int } > { 1 }
1506 {
1507 \skip_zero:N \topskip
1508 \skip_set_eq:NN \multicolsep \l__enumext_minipage_right_skip
1509 }
1510 }
1511 \cs_new_protected:Nn __enumext_keyans_minipage_add_space:
1512 {
1513 __enumext_keyans_minipage_set_skip:
1514 __enumext_unskip_unkern:
1515 \mode_if_vertical:TF
1516 {
1517 \nopagebreak\nointerlineskip
1518 }
1519 {
1520 \par\nopagebreak\nointerlineskip
1521 \skip_zero:N \l__enumext_partopsep_v_skip
1522 }
1523 \int_compare:nNnTF { \l__enumext_columns_v_int } > { 1 }
1524 {
1525 \addvspace{ 0.445\box_ht:N \strutbox }
1526 }
1527 {
1528 \addvspace{ 0.250\box_ht:N \strutbox }
1529 }
1530 }
1531 \cs_new_protected:Nn __enumext_keyans_pre_itemsep_skip:
1532 {
1533 \skip_if_eq:nnTF
1534 { \l__enumext_itemsep_i_skip } { \l__enumext_minipage_after_skip }
1535 {
1536 \skip_set:Nn \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1537 \skip_set:Nn \l__enumext_multicols_below_v_skip { 0.350\box_ht:N \strutbox }
1538 }
1539 {
1540 \dim_compare:nNnT
1541 { \l__enumext_itemsep_i_skip } < { \l__enumext_minipage_after_skip }
1542 {
1543 \skip_sub:Nn \l__enumext_minipage_after_skip { \l__enumext_itemsep_i_skip }
1544 \skip_sub:Nn \l__enumext_multicols_below_v_skip { \l__enumext_itemsep_i_skip }
1545 \skip_add:Nn \l__enumext_minipage_after_skip { 0.150\box_ht:N \strutbox }
1546 \skip_add:Nn \l__enumext_multicols_below_v_skip { 0.350\box_ht:N \strutbox }
1547 }
1548 \dim_compare:nNnT
1549 { \l__enumext_itemsep_i_skip } > { \l__enumext_minipage_after_skip }
1550 {
1551 \skip_set:Nn \l__enumext_minipage_temp_skip
1552 {
1553 \l__enumext_itemsep_i_skip - \l__enumext_minipage_after_skip
1554 }
1555 \skip_sub:Nn \l__enumext_minipage_after_skip { \l__enumext_itemsep_i_skip }
1556 \skip_sub:Nn \l__enumext_multicols_below_v_skip { \l__enumext_itemsep_i_skip }
1557 \skip_add:Nn \l__enumext_minipage_after_skip
1558 { 0.150\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }
1559 \skip_add:Nn \l__enumext_multicols_below_v_skip
1560 { 0.350\box_ht:N \strutbox + \l__enumext_minipage_temp_skip }

```



```

1561 }
1562 }
1563 }

```

(End of definition for `\__enumext_keyans_minipage_set_skip:`, `\__enumext_keyans_minipage_add_space:`, and `\__enumext_keyans_pre_itemsep_skip:`.)

### 13.23.3 Adjustment of vertical spaces for minipage in enumext\* and keyans\*

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_page` environment in `enumext*` and `keyans*`.

```

__enumext_mini_set_vskip_vii:
__enumext_mini_set_vskip_viii:

```

```

1564 \cs_new_protected:Nn __enumext_mini_set_vskip_vii:
1565 {
1566 \skip_zero_new:N \l__enumext_minipage_left_skip
1567 \skip_gzero_new:N \g__enumext_minipage_right_skip
1568 \skip_gzero_new:N \g__enumext_minipage_after_skip
1569 \skip_if_eq:nnTF { \l__enumext_topsep_vii_skip } { \c_zero_skip }
1570 {
1571 \skip_set:Nn \l__enumext_minipage_left_skip { 0.5\box_dp:N \strutbox }
1572 \skip_gset:Nn \g__enumext_minipage_right_skip { 0.325\box_dp:N \strutbox }
1573 }
1574 {
1575 \skip_set:Nn \l__enumext_minipage_left_skip { 0.5875\box_dp:N \strutbox }
1576 \skip_gset:Nn \g__enumext_minipage_right_skip
1577 {
1578 \l__enumext_topsep_vii_skip
1579 }
1580 \skip_gset:Nn \g__enumext_minipage_after_skip
1581 {
1582 0.325\box_dp:N \strutbox + \l__enumext_topsep_vii_skip
1583 }
1584 }
1585 }
1586 \cs_new_protected:Nn __enumext_mini_set_vskip_viii:
1587 {
1588 \skip_zero_new:N \l__enumext_minipage_after_skip
1589 \skip_zero_new:N \l__enumext_minipage_left_skip
1590 \skip_zero_new:N \l__enumext_minipage_right_skip
1591 \skip_if_eq:nnTF { \l__enumext_topsep_viii_skip } { \c_zero_skip }
1592 {
1593 \skip_set:Nn \l__enumext_minipage_left_skip
1594 {
1595 0.5\box_dp:N \strutbox
1596 }
1597 \skip_set:Nn \l__enumext_minipage_right_skip
1598 {
1599 \l__enumext_partopsep_viii_skip
1600 }
1601 \skip_set:Nn \l__enumext_minipage_after_skip
1602 {
1603 1.6\box_dp:N \strutbox
1604 }
1605 }
1606 {
1607 \skip_set:Nn \l__enumext_minipage_left_skip
1608 {
1609 0.5875\box_dp:N \strutbox
1610 }
1611 \skip_set:Nn \l__enumext_minipage_right_skip
1612 {
1613 \l__enumext_topsep_viii_skip
1614 }
1615 \skip_set:Nn \l__enumext_minipage_after_skip
1616 {
1617 0.325\box_dp:N \strutbox + \l__enumext_topsep_viii_skip
1618 }
1619 }
1620 }

```

(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_page` environment on the *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 `0pt` in both environments.

```

1621 \cs_new_protected:Nn __enumext_mini_addvspace_vii:
1622 {
1623 __enumext_mini_set_vskip_vii:
1624 \par\nopagebreak
1625 \addvspace { \l__enumext_minipage_left_skip }
1626 }
1627 \cs_new_protected:Nn __enumext_mini_addvspace_viii:
1628 {
1629 __enumext_mini_set_vskip_viii:
1630 \par\nopagebreak
1631 \addvspace { \l__enumext_minipage_left_skip }
1632 }

```

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

### 13.23.4 The command `\miniright`

The command `\miniright` will close the `\__enumext_mini_page` environment on the “left side”, open the `\__enumext_mini_page` 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  $\TeX$  justification is maintained in the `\__enumext_mini_page` on the “right side”.

`\miniright`

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

```

1633 \NewDocumentCommand \miniright { s }
1634 {
1635 \int_compare:nNt { \l__enumext_keyans_pic_level_int } = { 1 }
1636 {
1637 \msg_error:nnn { enumext } { wrong-miniright-place }
1638 }
1639 % outside
1640 \bool_lazy_and:nnT
1641 { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
1642 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
1643 {
1644 \msg_error:nnn { enumext } { wrong-miniright-place }
1645 }
1646 % starred env
1647 \bool_lazy_and:nnT
1648 { \bool_if_p:N \g__enumext_starred_bool }
1649 { \bool_not_p:n { \l__enumext_standar_bool } }
1650 {
1651 \msg_error:nnn { enumext } { wrong-miniright-starred }
1652 }
1653 % exec
1654 \int_compare:nNtF { \l__enumext_keyans_level_int } = { 1 }
1655 {
1656 __enumext_keyans_mini_right_cmd:n {#1}
1657 }
1658 { __enumext_mini_right_cmd:n {#1} }
1659 }

```

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

`\__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_page` environment on the “left side”, then we open the `\__enumext_mini_page` 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.

```

1660 \cs_new_protected:Npn __enumext_mini_right_cmd:n #1
1661 {
1662 \dim_compare:nNtF
1663 { \dim_use:c { \l__enumext_minipage_right_ __enumext_level: _dim } } > { \c_zero_dim }

```

```

1664 {
1665 __enumext_multicols_stop:
1666 \int_compare:nNnT
1667 { \int_use:c { __enumext_columns_ __enumext_level: _int } } = { 1 }
1668 {
1669 \par\addvspace{ __enumext_minipage_after_skip }
1670 }
1671 \end__enumext_mini_page
1672 \hfill
1673 __enumext_mini_page{ \dim_use:c { __enumext_minipage_right_ __enumext_level: _dim } }
1674 \par\nointerlineskip
1675 \addvspace { __enumext_minipage_right_skip }
1676 \bool_if:nF {#1}
1677 {
1678 \centering
1679 }
1680 \int_gzero:N \g__enumext_minipage_stat_int
1681 }
1682 { \msg_error:nnn { enumext } { wrong-miniright-use } }
1683 % paranoia
1684 \RenewDocumentCommand \miniright { s }
1685 {
1686 \msg_error:nn { enumext } { many-miniright-used }
1687 }
1688 }

```

(End of definition for \\_\_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.

```

1689 \cs_new_protected:Npn __enumext_keyans_mini_right_cmd:n #1
1690 {
1691 \dim_compare:nNnTF { __enumext_minipage_right_v_dim } > { \c_zero_dim }
1692 {
1693 __enumext_keyans_multicols_stop:
1694 \int_compare:nNnT { __enumext_columns_v_int } = { 1 }
1695 {
1696 \par\addvspace{ __enumext_minipage_after_skip }
1697 }
1698 \end__enumext_mini_page
1699 \hfill
1700 __enumext_mini_page{ __enumext_minipage_right_v_dim }
1701 \par\nointerlineskip
1702 \addvspace { __enumext_minipage_right_skip }
1703 \bool_if:nF {#1}
1704 {
1705 \centering
1706 }
1707 \int_gzero:N \g__enumext_minipage_stat_int
1708 }
1709 { \msg_error:nnn { enumext } { wrong-miniright-use } }
1710 % paranoia
1711 \RenewDocumentCommand \miniright { s }
1712 {
1713 \msg_error:nn { enumext } { many-miniright-used }
1714 }
1715 }

```

(End of definition for \\_\_enumext\_keyans\_mini\_right\_cmd:n.)

### 13.24 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 *⟨keys⟩* dedicated to this purpose, in this case it is best to use **\vspace** or **\vspace\*** when convenient.

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

```

above* 1716 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
below
below*

```

```

1717 {
1718 \keys_define:nn { enumext / #1 }
1719 {
1720 above .skip_set:c = { __enumext_vspace_above_#2_skip },
1721 above .value_required:n = true,
1722 above* .code:n = \bool_set_true:c { __enumext_vspace_a_star_#2_bool }
1723 \keys_set:nn { enumext / #1 } { above = {##1} },
1724 above* .value_required:n = true,
1725 below .skip_set:c = { __enumext_vspace_below_#2_skip },
1726 below .value_required:n = true,
1727 below* .code:n = \bool_set_true:c { __enumext_vspace_b_star_#2_bool }
1728 \keys_set:nn { enumext / #1 } { below = {##1} },
1729 below* .value_required:n = true,
1730 }
1731 }
1732 \clist_map_inline:Nn \c__enumext_all_envs_clist { __enumext_tmp:nn #1 }

```

(End of definition for *above* and *others*.)

### 13.24.1 Functions for *above* and *below* keys in *enumext*

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

```

1733 \cs_new_protected:Nn __enumext_vspace_above:
1734 {
1735 \skip_if_eq:nnF
1736 { \skip_use:c { __enumext_vspace_above_ __enumext_level: _skip } } { \c_zero_skip }
1737 {
1738 \bool_if:cTF { __enumext_vspace_a_star_ __enumext_level: _bool }
1739 {
1740 \vspace*{ \skip_use:c { __enumext_vspace_above_ __enumext_level: _skip } }
1741 }
1742 {
1743 \vspace { \skip_use:c { __enumext_vspace_above_ __enumext_level: _skip } }
1744 }
1745 }
1746 }

```

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

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

```

1747 \cs_new_protected:Nn __enumext_vspace_below:
1748 {
1749 \skip_if_eq:nnF
1750 { \skip_use:c { __enumext_vspace_below_ __enumext_level: _skip } } { \c_zero_skip }
1751 {
1752 \bool_if:cTF { __enumext_vspace_b_star_ __enumext_level: _bool }
1753 {
1754 \vspace*{ \skip_use:c { __enumext_vspace_below_ __enumext_level: _skip } }
1755 }
1756 {
1757 \vspace { \skip_use:c { __enumext_vspace_below_ __enumext_level: _skip } }
1758 }
1759 }
1760 }

```

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

### 13.24.2 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.

```

1761 \cs_new_protected:Nn __enumext_vspace_above_v:
1762 {
1763 \skip_if_eq:nnF { __enumext_vspace_above_v_skip } { \c_zero_skip }
1764 {
1765 \bool_if:NTF __enumext_vspace_a_star_v_bool
1766 {
1767 \vspace*{ __enumext_vspace_above_v_skip }
1768 }
1769 { \vspace { __enumext_vspace_above_v_skip } }
1770 }
1771 }

```

(End of definition for `\__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.

```

1772 \cs_new_protected:Nn __enumext_vspace_below_v:
1773 {
1774 \skip_if_eq:nnF { \l__enumext_vspace_below_v_skip } { \c_zero_skip }
1775 {
1776 \bool_if:NTF \l__enumext_vspace_b_star_v_bool
1777 {
1778 \vspace*{ \l__enumext_vspace_below_v_skip }
1779 }
1780 { \vspace { \l__enumext_vspace_below_v_skip } }
1781 }
1782 }

```

(End of definition for `\__enumext_vspace_below_v:`)

### 13.24.3 Functions for above and below keys in enumext\* keyans\*

`\__enumext_vspace_above_vii:`

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.

`\__enumext_vspace_above_viii:`

```

1783 \cs_new_protected:Nn __enumext_vspace_above_vii:
1784 {
1785 \skip_if_eq:nnF { \l__enumext_vspace_above_vii_skip } { \c_zero_skip }
1786 {
1787 \bool_if:NTF \l__enumext_vspace_a_star_vii_bool
1788 {
1789 \vspace*{ \l__enumext_vspace_above_vii_skip }
1790 }
1791 { \vspace { \l__enumext_vspace_above_vii_skip } }
1792 }
1793 }
1794 \cs_new_protected:Nn __enumext_vspace_above_viii:
1795 {
1796 \skip_if_eq:nnF { \l__enumext_vspace_above_viii_skip } { \c_zero_skip }
1797 {
1798 \bool_if:NTF \l__enumext_vspace_a_star_viii_bool
1799 {
1800 \vspace*{ \l__enumext_vspace_above_viii_skip }
1801 }
1802 { \vspace { \l__enumext_vspace_above_viii_skip } }
1803 }
1804 }

```

(End of definition for `\__enumext_vspace_above_vii:` and `\__enumext_vspace_above_viii:`)

`\__enumext_vspace_below_vii:`

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.

`\__enumext_vspace_below_viii:`

```

1805 \cs_new_protected:Nn __enumext_vspace_below_vii:
1806 {
1807 \skip_if_eq:nnF { \l__enumext_vspace_below_vii_skip } { \c_zero_skip }
1808 {
1809 \bool_if:NTF \l__enumext_vspace_b_star_vii_bool
1810 {
1811 \vspace*{ \l__enumext_vspace_below_vii_skip }
1812 }
1813 { \vspace { \l__enumext_vspace_below_vii_skip } }
1814 }
1815 }
1816 \cs_new_protected:Nn __enumext_vspace_below_viii:
1817 {
1818 \skip_if_eq:nnF { \l__enumext_vspace_below_viii_skip } { \c_zero_skip }
1819 {
1820 \bool_if:NTF \l__enumext_vspace_b_star_viii_bool
1821 {
1822 \vspace*{ \l__enumext_vspace_below_viii_skip }
1823 }
1824 { \vspace { \l__enumext_vspace_below_viii_skip } }
1825 }
1826 }

```

(End of definition for `\__enumext_vspace_below_vii:` and `\__enumext_vspace_below_viii:`)

### 13.25 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 `<keys>` passed to the *optional argument* of the “first level” of the environments `enumext` and `enumext*`, but, discarding some specific `<keys>`. 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
resume*
1827 \cs_set_protected:Npn __enumext_tmp:n #1
1828 {
1829 \keys_define:nn { enumext / #1 }
1830 {
1831 series .str_set:N = __enumext_series_str,
1832 series .value_required:n = true,
1833 resume .code:n = __enumext_resume_series:n {##1},
1834 resume* .code:n = __enumext_resume_starred:,
1835 resume* .value_forbidden:n = true,
1836 }
1837 }
1838 \clist_map_inline:nn { level-1, enumext* } { __enumext_tmp:n {#1} }

```

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

#### 13.25.1 Internal functions for series key

The function `\__enumext_filter_series:n` will be in charge of filtering the `<keys>` we want to store where `{#1}` represents the *optional argument* passed to the environment.

```

__enumext_filter_series:n
 __enumext_filter_series_key:n
 __enumext_filter_series_pair:nn
1839 \cs_new:Npn __enumext_filter_series:n #1
1840 {
1841 \use:e
1842 {
1843 \keyval_parse:NNn
1844 __enumext_filter_series_key:n
1845 __enumext_filter_series_pair:nn {#1}
1846 }
1847 }

```

The function `\__enumext_filter_series_key:n` will be responsible for filtering the `<keys>` that are passed “without value” by excluding the `resume`, `resume*` and `base-fix` keys.

```

1848 \cs_new:Npn __enumext_filter_series_key:n #1
1849 {
1850 \str_case:nnF {#1}
1851 {
1852 { resume } {} { resume* } {} { base-fix } {}
1853 }
1854 { , { \exp_not:n {#1} } }
1855 }

```

The function `\__enumext_filter_series_pair:nn` will be responsible for filtering the `<keys>` that are passed “with value” by excluding the `series`, `resume`, `start`, `start*`, `save-ans` and `save-key` keys.

```

1856 \cs_new:Npn __enumext_filter_series_pair:nn #1#2
1857 {
1858 \str_case:nnF {#1}
1859 {
1860 { series } {} { resume } {} { start } {}
1861 { start* } {} { save-ans } {} { save-key } {}
1862 }
1863 { , { \exp_not:n {#1} } = { \exp_not:n {#2} } }
1864 }

```

(End of definition for `\__enumext_filter_series:n`, `\__enumext_filter_series_key:n`, and `\__enumext_filter_series_pair:nn`.)

The function `\__enumext_parse_series:n` will be responsible for storing the filtered `<keys>` in the global variable `\g__enumext_series_<series name>_tl` along with the creation of the integer variable `\g__enumext_series_<series name>_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 `<keys>`. This function is passed to the function `\__enumext_parse_keys:n` in the `enumext` environment definition (§13.39) and to the function `\__enumext_parse_keys_vii:n` in the `enumext*` environment definition (§13.44).



```

1865 \cs_new_protected:Npn __enumext_parse_series:n #1
1866 {
1867 \str_if_empty:NTF \l__enumext_series_str
1868 {
1869 \bool_if:NF \l__enumext_resume_active_bool
1870 {
1871 __enumext_resume_last:n {#1}
1872 }
1873 }
1874 {
1875 \tl_gclear_new:c { g__enumext_series_ \l__enumext_series_str _tl }
1876 \tl_gset:ce { g__enumext_series_ \l__enumext_series_str _tl }
1877 { __enumext_filter_series:n {#1} }
1878 \int_if_exist:cF { g__enumext_series_ \l__enumext_series_str _int }
1879 {
1880 \int_new:c { g__enumext_series_ \l__enumext_series_str _int }
1881 }
1882 }
1883 }

```

The function `\__enumext_resume_last:n` will be in charge of saving the filtering (*keys*) when the *series* key is *not used* and will save them in the variable `\g__enumext_standar_series_tl` for the *enumext* environment and in the variable `\g__enumext_starred_series_tl` for the *enumext\** environment.

```

1884 \cs_new_protected:Npn __enumext_resume_last:n #1
1885 {
1886 \bool_if:NT \l__enumext_standar_first_bool
1887 {
1888 \tl_gclear:N \g__enumext_standar_series_tl
1889 \tl_gset:Ne \g__enumext_standar_series_tl { __enumext_filter_series:n {#1} }
1890 }
1891 \bool_if:NT \l__enumext_starred_first_bool
1892 {
1893 \tl_gclear:N \g__enumext_starred_series_tl
1894 \tl_gset:Ne \g__enumext_starred_series_tl { __enumext_filter_series:n {#1} }
1895 }
1896 }

```

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

### 13.25.2 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_⟨series name⟩_int` if the *series*={⟨series name⟩} 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_⟨series name⟩_int` if the key *resume*={⟨series name⟩} has been passed and in `\g__enumext_series_⟨store name⟩_int` if the key has been passed *save-ans*={⟨store name⟩}.

- 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 (§13.39) and the *enumext\** environment definition (§13.44).

```

1897 \cs_new_protected:Npn __enumext_resume_save_counter:
1898 {
1899 \bool_if:NT \g__enumext_standar_bool
1900 {
1901 \tl_if_empty:NF \l__enumext_series_str
1902 {
1903 \int_gset_eq:cN
1904 { g__enumext_series_ \l__enumext_series_str _int } \value{enumXi}
1905 }
1906 \tl_if_empty:NTF \l__enumext_resume_name_tl
1907 {
1908 \str_if_empty:NF \l__enumext_series_str
1909 {
1910 \int_gset_eq:NN \g__enumext_resume_int \value{enumXi}
1911 }
1912 }
1913 {
1914 \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
1915 {
1916 \int_gset_eq:cN

```

```

1917 { g__enumext_series_ \l__enumext_resume_name_tl _int } \value{enumXi}
1918 }
1919 }
1920 \int_if_exist:cT { g__enumext_resume_ \l__enumext_store_name_tl _int }
1921 {
1922 \int_gset_eq:cN
1923 { g__enumext_resume_ \l__enumext_store_name_tl _int } \value{enumXi}
1924 }
1925 }
1926 \bool_if:NT \g__enumext_starred_bool
1927 {
1928 \tl_if_empty:NF \l__enumext_series_str
1929 {
1930 \int_gset_eq:cN
1931 { g__enumext_series_ \l__enumext_series_str _int } \value{enumXvii}
1932 }
1933 \tl_if_empty:NTF \l__enumext_resume_name_tl
1934 {
1935 \str_if_empty:NT \l__enumext_series_str
1936 {
1937 \int_gset_eq:NN \g__enumext_resume_vii_int \value{enumXvii}
1938 }
1939 }
1940 {
1941 \int_if_exist:cT { g__enumext_series_ \l__enumext_resume_name_tl _int }
1942 {
1943 \int_gset_eq:cN
1944 { g__enumext_series_ \l__enumext_resume_name_tl _int } \value{enumXvii}
1945 }
1946 }
1947 \int_if_exist:cT { g__enumext_resume_ \l__enumext_store_name_tl _int }
1948 {
1949 \int_gset_eq:cN
1950 { g__enumext_resume_ \l__enumext_store_name_tl _int } \value{enumXvii}
1951 }
1952 }
1953 }

```

(End of definition for \\_\_enumext\_resume\_save\_counter:.)

### 13.25.3 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={⟨series name⟩}` 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_⟨series name⟩_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.

```

1954 \cs_new_protected:Npn __enumext_resume_series:n #1
1955 {
1956 \tl_if_empty:NTF {#1}
1957 {
1958 __enumext_resume_counter:n { }
1959 }
1960 {
1961 \tl_if_exist:cTF { g__enumext_series_ \tl_to_str:n {#1} _tl }
1962 {
1963 __enumext_resume_counter:n {#1}
1964 \bool_if:NT \g__enumext_standar_bool
1965 {
1966 \keys_set:nv { enumext / level-1 }
1967 { g__enumext_series_ \tl_to_str:n {#1} _tl }
1968 }
1969 \bool_if:NT \g__enumext_starred_bool
1970 {
1971 \keys_set:nv { enumext / enumext* }
1972 { g__enumext_series_ \tl_to_str:n {#1} _tl }
1973 }
1974 }
1975 {
1976 \bool_if:NT \g__enumext_standar_bool

```

```

1977 {
1978 \msg_error:nnn { enumext } { unknown-series } {#1}
1979 }
1980 \bool_if:NT \g__enumext_starred_bool
1981 {
1982 \msg_error:nnn { enumext } { unknown-series } {#1}
1983 }
1984 }
1985 }
1986 }

```

(End of definition for \\_\_enumext\_resume\_series:n)

```

__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`.

```

1987 \cs_new_protected:Npn __enumext_resume_counter:n #1
1988 {
1989 \bool_set_true:N \l__enumext_resume_active_bool
1990 \tl_set:Nn \l__enumext_resume_name_tl {#1}
1991 \tl_if_empty:NTF \l__enumext_resume_name_tl
1992 {
1993 __enumext_resume_counter:
1994 }
1995 {
1996 __enumext_resume_counter_series:
1997 }
1998 __enumext_resume_counter_save_ans:
1999 }

```

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.

```

2000 \cs_new_protected:Nn __enumext_resume_counter:
2001 {
2002 \bool_if:NT \g__enumext_standar_bool
2003 {
2004 \int_gincr:N \g__enumext_resume_int
2005 \int_set_eq:NN \l__enumext_start_i_int \g__enumext_resume_int
2006 }
2007 \bool_if:NT \g__enumext_starred_bool
2008 {
2009 \int_gincr:N \g__enumext_resume_vii_int
2010 \int_set_eq:NN \l__enumext_start_vii_int \g__enumext_resume_vii_int
2011 }
2012 }

```

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.

```

2013 \cs_new_protected:Nn __enumext_resume_counter_series:
2014 {
2015 \bool_if:NT \g__enumext_standar_bool
2016 {
2017 \int_set:Nn \l__enumext_start_i_int
2018 {
2019 \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
2020 }
2021 }
2022 \bool_if:NT \g__enumext_starred_bool
2023 {
2024 \int_set:Nn \l__enumext_start_vii_int
2025 {
2026 \int_use:c { g__enumext_series_ \l__enumext_resume_name_tl _int } + 1
2027 }
2028 }
2029 }

```

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.

```

2030 \cs_new_protected:Nn __enumext_resume_counter_save_ans:
2031 {
2032 \bool_lazy_and:nnT
2033 { \bool_if_p:N __enumext_standar_first_bool }
2034 { \bool_if_p:N __enumext_store_active_bool }
2035 {
2036 \int_set:Nn __enumext_start_i_int
2037 {
2038 \int_use:c { g__enumext_resume_ __enumext_store_name_tl _int } + 1
2039 }
2040 }
2041 \bool_lazy_and:nnT
2042 { \bool_if_p:N __enumext_starred_first_bool }
2043 { \bool_if_p:N __enumext_store_active_bool }
2044 {
2045 \int_set:Nn __enumext_start_vii_int
2046 {
2047 \int_use:c { g__enumext_resume_ __enumext_store_name_tl _int } + 1
2048 }
2049 }
2050 }

```

(End of definition for `\__enumext_resume_counter:n` and others.)

### 13.25.4 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 `<keys>` 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={<series name>}` or `series={<series name>}` were not active.

```

2051 \cs_new_protected:Nn __enumext_resume_starred:
2052 {
2053 \bool_if:NT \g__enumext_standar_bool
2054 {
2055 \tl_if_empty:NF \g__enumext_standar_series_tl
2056 {
2057 __enumext_resume_counter:n { }
2058 \keys_set:nV { enumext / level-1 } \g__enumext_standar_series_tl
2059 }
2060 }
2061 \bool_if:NT \g__enumext_starred_bool
2062 {
2063 \tl_if_empty:NF \g__enumext_starred_series_tl
2064 {
2065 __enumext_resume_counter:n { }
2066 \keys_set:nV { enumext / enumext* } \g__enumext_starred_series_tl
2067 }
2068 }
2069 }

```

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

## 13.26 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.

### 13.26.1 Setting `save-ans` key

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

```

2070 \cs_set_protected:Npn __enumext_tmp:n #1
2071 {
2072 \keys_define:nn { enumext / #1 }
2073 {
2074 save-ans .code:n = __enumext_storing_set:n {##1},
2075 save-ans .value_required:n = true,
2076 }
2077 }
2078 \clist_map_inline:nn { level-1, enumext* } { { __enumext_tmp:n {#1} } }

```

(End of definition for `save-ans`.)

### 13.26.2 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:`.

```

2079 \cs_new_protected:Nn __enumext_start_save_ans_msg:
2080 {
2081 \msg_term:nnVV { enumext } { save-ans-log }
2082 \g__enumext_envir_name_tl \l__enumext_store_name_tl
2083 }
2084 \cs_new_protected:Nn __enumext_stop_save_ans_msg:
2085 {
2086 \msg_term:nnVV { enumext } { save-ans-log-hook }
2087 \g__enumext_envir_name_tl \g__enumext_store_name_tl
2088 }

```

(End of definition for `\__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 *sequence* and *prop list* 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.

```

2089 \cs_new_protected:Npn __enumext_storing_set:n #1
2090 {
2091 \tl_set:Nx \l__enumext_store_name_tl {#1}
2092 \tl_if_empty:NTF \l__enumext_store_name_tl
2093 {
2094 \bool_lazy_or:nnT
2095 { \l__enumext_standar_first_bool } { \l__enumext_starred_first_bool }
2096 {
2097 \msg_error:nnV { enumext } { save-ans-empty } \g__enumext_envir_name_tl
2098 }
2099 }
2100 {
2101 \bool_lazy_or:nnT
2102 { \l__enumext_standar_first_bool } { \l__enumext_starred_first_bool }
2103 {
2104 __enumext_start_save_ans_msg:
2105 __enumext_storing_exec:
2106 }
2107 }
2108 }

```

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 `anskey*`, `keyans`, `keyans*` and `keyanspic` environments and will set to “true” the variable `\l__enumext_check_answers_bool` used for internal checking answers mechanism set by the `check-ans` and `no-store` keys, copy `{⟨store name⟩}` into the variable `\g__enumext_store_name_tl` and execute the function `\__enumext_anskey_env_make:V` creating the environment `anskey*` (§13.31).

```

2109 \cs_new_protected:Nn __enumext_storing_exec:
2110 {
2111 \bool_set_true:N \l__enumext_store_active_bool
2112 \bool_set_true:N \l__enumext_check_answers_bool
2113 \tl_gset:NV \g__enumext_store_name_tl \l__enumext_store_name_tl
2114 __enumext_anskey_env_make:V \l__enumext_store_name_tl

```

The *prop list* `\g__enumext_series_⟨store name⟩_prop` and the *sequence* `\g__enumext_series_⟨store name⟩_seq` will be created globally to “*store content*” in case they do not exist together with the integer variable `\g__enumext_series_⟨store name⟩_int` used by the keys `resume` and `resume*`.

```

2115 \prop_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _prop }
2116 {
2117 \msg_log:nnV { enumext } { store-prop } \l__enumext_store_name_tl
2118 \prop_new:c { g__enumext_ \l__enumext_store_name_tl _prop }
2119 }
2120 \seq_if_exist:cF { g__enumext_ \l__enumext_store_name_tl _seq }
2121 {
2122 \msg_log:nnV { enumext } { store-seq } \l__enumext_store_name_tl
2123 \seq_new:c { g__enumext_ \l__enumext_store_name_tl _seq }

```

```

2124 }
2125 \int_if_exist:cF { g__enumext_resume_ \l__enumext_store_name_tl _int }
2126 {
2127 \msg_log:nnV { enumext } { store-int } \l__enumext_store_name_tl
2128 \int_new:c { g__enumext_resume_ \l__enumext_store_name_tl _int }
2129 }
2130 }

```

(End of definition for `\__enumext_storing_set:n` and `\__enumext_storing_exec:.`)

### 13.26.3 The check answer mechanism

The internal mechanism for “checking answers” 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.

### 13.26.4 Setting check-ans and no-store keys

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

```

check-ans 2131 \cs_set_protected:Npn __enumext_tmp:n #1
no-store 2132 {
2133 \keys_define:nn { enumext / #1 }
2134 {
2135 check-ans .bool_set:N = \l__enumext_check_ans_key_bool,
2136 check-ans .initial:n = false,
2137 check-ans .value_required:n = true,
2138 no-store .code:n = {
2139 \bool_set_false:N \l__enumext_check_answers_bool
2140 \bool_set_false:N \l__enumext_check_ans_key_bool
2141 },
2142 no-store .value_forbidden:n = true,
2143 }
2144 }
2145 \clist_map_inline:nn
2146 {
2147 level-1, level-2, level-3, level-4, enumext*
2148 }
2149 { __enumext_tmp:n {#1} }

```

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

### 13.26.5 Set-up check answer mechanism

`\__enumext_check_ans_active:` 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.

```

2150 \cs_new_protected:Nn __enumext_check_ans_active:
2151 {
2152 \tl_if_empty:NF \l__enumext_store_name_tl
2153 {
2154 \bool_if:NT \l__enumext_check_answers_bool
2155 {
2156 __enumext_check_ans_level:
2157 }
2158 }
2159 }

```

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*”.

```

2160 \cs_new_protected:Nn __enumext_check_ans_level:
2161 {
2162 \int_case:nn { \l__enumext_level_int }
2163 {
2164 { 1 }{
2165 \bool_lazy_all:nT
2166 {
2167 { \bool_if_p:N \g__enumext_starred_bool }
2168 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 1 } }
2169 }
2170 {
2171 \int_gdecr:N \g__enumext_item_number_int
2172 \bool_set_false:N \l__enumext_item_number_bool
2173 }
2174 }
2175 { 2 }{
2176 \int_gdecr:N \g__enumext_item_number_int
2177 \bool_set_false:N \l__enumext_item_number_bool
2178 }
2179 { 3 }{
2180 \int_gdecr:N \g__enumext_item_number_int
2181 \bool_set_false:N \l__enumext_item_number_bool
2182 }
2183 { 4 }{
2184 \int_gdecr:N \g__enumext_item_number_int
2185 \bool_set_false:N \l__enumext_item_number_bool
2186 }
2187 }

```

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.

```

2188 \int_case:nn { \l__enumext_level_h_int }
2189 {
2190 { 1 }{
2191 \bool_lazy_all:nT
2192 {
2193 { \bool_if_p:N \g__enumext_standar_bool }
2194 { \int_compare_p:nNn { \l__enumext_level_int } = { 1 } }
2195 }
2196 {
2197 \int_gdecr:N \g__enumext_item_number_int
2198 \bool_set_false:N \l__enumext_item_number_bool
2199 }
2200 }
2201 }
2202 }

```

(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 \l\_\_enumext\_check\_ans\_key\_bool to the global variable \g\_\_enumext\_check\_ans\_key\_bool only if the key `check-ans` is active.

```

2203 \cs_new_protected:Nn __enumext_check_ans_key_hook:
2204 {
2205 \bool_lazy_and:nnT
2206 { \bool_if_p:N \l__enumext_check_ans_key_bool }
2207 { \bool_if_p:N \g__enumext_standar_bool }
2208 {
2209 \bool_gset_true:N \g__enumext_check_ans_key_bool
2210 }
2211 \bool_lazy_and:nnT
2212 { \bool_if_p:N \l__enumext_check_ans_key_bool }
2213 { \bool_if_p:N \g__enumext_starred_bool }
2214 {
2215 \bool_gset_true:N \g__enumext_check_ans_key_bool
2216 }
2217 }

```

(End of definition for \\_\_enumext\_check\_ans\_key\_hook:.)

\\_\_enumext\_item\_answer\_diff:

The function \\_\_enumext\_item\_answer\_diff: will set the value of the variable \g\_\_enumext\_item\_answer\_diff\_int which is used by the functions \\_\_enumext\_check\_ans\_show: for the key `save-ans` 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:.

```

2218 \cs_new_protected:Nn __enumext_item_answer_diff:
2219 {
2220 \int_gset:Nn \g__enumext_item_answer_diff_int
2221 {
2222 \int_sign:n { \g__enumext_item_number_int - \g__enumext_item_anskey_int }
2223 }
2224 }

```

(End of definition for \\_\_enumext\_item\_answer\_diff:.)

\\_\_enumext\_check\_ans\_show:

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:.

```

2225 \cs_new_protected:Nn __enumext_check_ans_show:
2226 {
2227 \int_case:nn { \g__enumext_item_answer_diff_int }
2228 {
2229 { -1 } { __enumext_check_ans_msg_less: }
2230 { 0 } { __enumext_check_ans_msg_same_ok: }
2231 { 1 } { __enumext_check_ans_msg_greater: }
2232 }
2233 }
2234 \cs_new_protected:Nn __enumext_check_ans_msg_less:
2235 {
2236 \msg_warning:nneee { enumext } { item-less-answer } { \g__enumext_store_name_tl }
2237 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2238 }
2239 \cs_new_protected:Nn __enumext_check_ans_msg_same_ok:
2240 {
2241 \msg_term:nneee { enumext } { items-same-answer } { \g__enumext_store_name_tl }
2242 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2243 }
2244 \cs_new_protected:Nn __enumext_check_ans_msg_greater:
2245 {
2246 \msg_warning:nneee { enumext } { item-greater-answer } { \g__enumext_store_name_tl }
2247 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2248 }

```

(End of definition for \\_\_enumext\_check\_ans\_show: and others.)

\\_\_enumext\_check\_ans\_log:

The function \\_\_enumext\_check\_ans\_log: will be executed within the function \\_\_enumext\_execute\_after\_env: when the key `check-ans` is not active, that is, when \g\_\_enumext\_check\_ans\_key\_bool is “*false*” and write in the log the appropriate message according to the value of \g\_\_enumext\_item\_answer\_diff\_int set by the function \\_\_enumext\_item\_answer\_diff:.

\\_\_enumext\_check\_ans\_log\_msg\_less:

\\_\_enumext\_check\_ans\_log\_msg\_same\_ok:

\\_\_enumext\_check\_ans\_log\_msg\_greater:

```

2249 \cs_new_protected:Nn __enumext_check_ans_log:
2250 {
2251 \int_case:nn { \g__enumext_item_answer_diff_int }
2252 {
2253 { -1 } { __enumext_check_ans_log_msg_less: }
2254 { 0 } { __enumext_check_ans_log_msg_same_ok: }
2255 { 1 } { __enumext_check_ans_log_msg_greater: }
2256 }
2257 }
2258 \cs_new_protected:Nn __enumext_check_ans_log_msg_less:
2259 {
2260 \msg_log:nneee { enumext } { item-less-answer } { \g__enumext_store_name_tl }
2261 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2262 }
2263 \cs_new_protected:Nn __enumext_check_ans_log_msg_same_ok:
2264 {
2265 \msg_log:nneee { enumext } { items-same-answer } { \g__enumext_store_name_tl }
2266 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2267 }
2268 \cs_new_protected:Nn __enumext_check_ans_log_msg_greater:
2269 {
2270 \msg_log:nneee { enumext } { item-greater-answer } { \g__enumext_store_name_tl }
2271 { \g__enumext_envir_name_tl } { \g__enumext_start_line_tl }
2272 }

```

(End of definition for \\_\_enumext\_check\_ans\_log: and others.)

### 13.26.6 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.

```

2273 \cs_new_protected:Npn __enumext_check_starred_cmd:n #1
2274 {
2275 \int_compare:nNnT
2276 { \g__enumext_check_starred_cmd_int } = { 0 }
2277 {
2278 \msg_warning:nnnV
2279 { enumext } { missing-starred } { #1 } \l__enumext_check_start_line_env_tl
2280 }
2281 \int_compare:nNnT
2282 { \g__enumext_check_starred_cmd_int } > { 1 }
2283 {
2284 \msg_warning:nnnV
2285 { enumext } { many-starred } { #1 } \l__enumext_check_start_line_env_tl
2286 }
2287 \int_gzero:N \g__enumext_check_starred_cmd_int
2288 \tl_clear:N \l__enumext_check_start_line_env_tl
2289 }

```

(End of definition for \\_\_enumext\_check\_starred\_cmd:n.)

## 13.27 Keys and functions associated with storage

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

```

2290 \cs_set_protected:Npn __enumext_tmp:n #1
2291 {
2292 \keys_define:nn { enumext / #1 }
2293 {
2294 wrap-ans .cs_set_protected:Np = __enumext_anskey_wrapper:n ##1,
2295 wrap-ans .initial:n =
2296 {
2297 \fbox{\parbox[t]{\dimeval{\itemwidth -2\fboxsep -2\fboxrule}}{##1}}
2298 },
2299 wrap-ans .value_required:n = true,
2300 wrap-opt .cs_set_protected:Np = __enumext_keyans_wrapper_opt:n ##1,
2301 wrap-opt .initial:n = [{##1}],
2302 wrap-opt .value_required:n = true,
2303 wrap-key .cs_set_protected:Np = __enumext_keyans_wrapper_item:n ##1,
2304 wrap-key .value_required:n = true,

```

```

2305 save-sep .tl_set:N = \l__enumext_store_keyans_item_opt_sep_tl,
2306 save-sep .initial:n = {, ~ },
2307 save-sep .value_required:n = true,
2308 mark-ans .tl_set:N = \l__enumext_mark_answer_sym_tl,
2309 mark-ans .initial:n = \textasteriskcentered,
2310 mark-ans .value_required:n = true,
2311 mark-pos .choice:,
2312 mark-pos / left .code:n = \str_set:Nn \l__enumext_mark_position_str { l },
2313 mark-pos / right .code:n = \str_set:Nn \l__enumext_mark_position_str { r },
2314 mark-pos / unknown .code:n =
2315 \msg_error:nnee { enumext } { unknown-choice }
2316 { mark-pos } { left, ~ right } { \exp_not:n {##1} },
2317 mark-pos .initial:n = right,
2318 mark-pos .value_required:n = true,
2319 show-ans .bool_set:N = \l__enumext_show_answer_bool,
2320 show-ans .initial:n = false,
2321 show-ans .value_required:n = true,
2322 show-pos .bool_set:N = \l__enumext_show_position_bool,
2323 show-pos .initial:n = false,
2324 show-pos .value_required:n = true,
2325 mark-ref .tl_set:N = \l__enumext_mark_ref_sym_tl,
2326 mark-ref .initial:n = \textreferencemark,
2327 mark-ref .value_required:n = true,
2328 save-ref .bool_set:N = \l__enumext_store_ref_key_bool,
2329 save-ref .initial:n = false,
2330 save-ref .value_required:n = true,
2331 }
2332 }
2333 \clist_map_inline:nn { level-1, enumext* } { __enumext_tmp:n {##1} }

```

(End of definition for `wrap-ans` and others.)

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

```

mark-pos 2334 \cs_set_protected:Npn __enumext_tmp:n #1
show-ans 2335 {
show-pos 2336 \keys_define:nn { enumext / #1 }
2337 {
2338 mark-pos .choice:,
2339 mark-pos / left .code:n = \str_set:Nn \l__enumext_mark_position_str { l },
2340 mark-pos / right .code:n = \str_set:Nn \l__enumext_mark_position_str { r },
2341 mark-pos .initial:n = right,
2342 mark-pos .value_required:n = true,
2343 show-ans .bool_set:N = \l__enumext_show_answer_bool,
2344 show-ans .initial:n = false,
2345 show-ans .value_required:n = true,
2346 show-pos .bool_set:N = \l__enumext_show_position_bool,
2347 show-pos .initial:n = false,
2348 show-pos .value_required:n = true,
2349 }
2350 }
2351 \clist_map_inline:nn { keyans, keyans* } { __enumext_tmp:n {##1} }

```

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

### 13.27.1 Storing structure of the environments

The idea behind “*storing structure*” in the *sequence* 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 argument* passed to the levels of the environment in which it is executed and “*storing*” this in the *sequence*.

```

__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 the “*storing keys*” filtered from the *optional argument* 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 *⟨keys⟩*.

```

2352 \cs_new_protected:Npn __enumext_store_active_keys:n #1
2353 {
2354 \bool_if:cF { \l__enumext_store_save_key_ __enumext_level: _bool }
2355 {
2356 \tl_clear:c { \l__enumext_save_key_ __enumext_level: _tl }
2357 \tl_set:ce

```

```

2358 { __enumext_store_save_key_ __enumext_level: _tl }
2359 { __enumext_filter_save_key:n {#1} }
2360 }
2361 }
2362 \cs_new_protected:Npn __enumext_store_active_keys_vii:n #1
2363 {
2364 \bool_if:NF __enumext_store_save_key_vii_bool
2365 {
2366 \tl_clear:N __enumext_store_save_key_vii_tl
2367 \tl_set:Ne __enumext_store_save_key_vii_tl { __enumext_filter_save_key:n {#1} }
2368 }
2369 }

```

(End of definition for \\_\_enumext\_store\_active\_keys:n and \\_\_enumext\_store\_active\_keys\_vii:n.)

### 13.27.2 Setting save-key key

Since this “*storing structure*” in the *sequence* established by the `save-ans` key when executing `\anskey` or `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 “*storing structure*” in the *sequence*.

save-key

The values set by this key passed in the *optional argument* of the `enumext` and `enumext*` environments will override the values of the `\__enumext_store_save_key_X_tl` variable set by the functions `\__enumext_store_active_keys:n` and `\__enumext_store_active_keys_vii:n`. Now define the key `save-key` for all levels of `enumext` and `enumext*` environments.

```

2370 \cs_set_protected:Npn __enumext_tmp:n #1
2371 {
2372 \keys_define:nn { enumext / enumext* }
2373 {
2374 save-key .code:n = __enumext_parse_save_key_vii:n {##1},
2375 save-key .value_required:n = true,
2376 }
2377 \keys_define:nn { enumext / #1 }
2378 {
2379 save-key .code:n = __enumext_parse_save_key:n {##1},
2380 save-key .value_required:n = true,
2381 }
2382 }
2383 \clist_map_inline:nn { level-1, level-2, level-3, level-4 } { __enumext_tmp:n {#1} }

```

(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 keys*” in the variable `\__enumext_store_save_key_X_tl` for `enumext` and `enumext*`.

```

2384 \cs_new_protected:Npn __enumext_parse_save_key:n #1
2385 {
2386 \bool_set_true:c { __enumext_store_save_key_ __enumext_level: _bool }
2387 \tl_clear:c { __enumext_store_save_key_ __enumext_level: _tl }
2388 \tl_set:ce
2389 { __enumext_store_save_key_ __enumext_level: _tl }
2390 { __enumext_filter_save_key:n {#1} }
2391 }
2392 \cs_new_protected:Npn __enumext_parse_save_key_vii:n #1
2393 {
2394 \bool_set_true:N __enumext_store_save_key_vii_bool
2395 \tl_clear:N __enumext_store_save_key_vii_tl
2396 \tl_set:Ne __enumext_store_save_key_vii_tl { __enumext_filter_save_key:n {#1} }
2397 }

```

(End of definition for \\_\_enumext\_parse\_save\_key:n and \\_\_enumext\_parse\_save\_key\_vii:n.)

### 13.27.3 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 keys*” we want to *stored in sequence* where `{#1}` represents the *optional argument* passed to the environment.

```

2398 \cs_new:Npn __enumext_filter_save_key:n #1
2399 {
2400 \use:e
2401 {
2402 \keyval_parse:NNn
2403 __enumext_filter_save_key_key:n
2404 __enumext_filter_save_key_pair:nn {#1}
2405 }
2406 }

```

The function `\__enumext_filter_save_key_key:n` will be responsible for “*filtering keys*” that are passed “*without value*” by excluding the `resume`, `resume*`, `no-store` and `base-fix` keys.

```

2407 \cs_new:Npn __enumext_filter_save_key_key:n #1
2408 {
2409 \str_case:nnF {#1}
2410 {
2411 { resume } {} { resume* } {} { no-store } {} { base-fix } {}
2412 }
2413 { , { \exp_not:n {#1} } }
2414 }

```

The function `\__enumext_filter_save_key_pair:nn` will be responsible for “*filtering keys*” 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.

```

2415 \cs_new:Npn __enumext_filter_save_key_pair:nn #1#2
2416 {
2417 \str_case:nnF {#1}
2418 {
2419 { series } {} { resume } {} { save-ans } {} { save-ref } {}
2420 { save-key } {} { check-ans } {} { show-ans } {} { show-pos } {}
2421 { wrap-ans } {} { mark-ans } {} { wrap-opt } {} { save-sep } {}
2422 { mark-ref } {} { mini-env } {} { mini-sep } {} { mini-right } {}
2423 { mini-right* } {}
2424 }
2425 { , { \exp_not:n {#1} } } = { \exp_not:n {#2} } }
2426 }

```

(End of definition for `\__enumext_filter_save_key:n`, `\__enumext_filter_save_key_key:n`, and `\__enumext_filter_save_key_pair:nn`.)

### 13.27.4 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  $\{\langle content \rangle\}$  in *prop list* defined by `save-ans` key. The “*stored content*” is retrieved by means of the `\getkeyans` command.

The form in which the  $\{\langle content \rangle\}$  is “*stored*” in the *prop list* 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.

```

2427 \cs_new_protected:Npn __enumext_store_addto_prop:n #1
2428 {
2429 \prop_gput_if_not_in:cen { g__enumext_ \l__enumext_store_name_tl _prop }
2430 {
2431 \int_eval:n { \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } + 1 }
2432 }
2433 { #1 }
2434 }
2435 \cs_generate_variant:Nn __enumext_store_addto_prop:n { V }

```

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

### 13.27.5 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  $\{\langle content \rangle\}$  in *sequence* 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  $\{\langle content \rangle\}$  is stored in *sequence* 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.

```

2436 \cs_new_protected:Npn __enumext_store_addto_seq:n #1
2437 {
2438 \seq_gput_right:cn { g__enumext_ \l__enumext_store_name_tl _seq } { #1 }
2439 }
2440 \cs_generate_variant:Nn __enumext_store_addto_seq:n { v, V }

```

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

### 13.27.6 Functions for storing structure in the sequence

The “*storing structure*” is handled by the functions `\__enumext_store_level_open:` and `\__enumext_store_level_close:` which are executed per level within the `enumext` environment.

```

2441 \cs_new_protected:Nn __enumext_store_level_open:
2442 {
2443 \bool_if:NT \l__enumext_check_answers_bool
2444 {
2445 \tl_if_empty:cTF { l__enumext_store_save_key_ __enumext_level: _tl }
2446 {
2447 __enumext_store_addto_seq:n
2448 {
2449 \item \begin{enumext}
2450 }
2451 }
2452 {
2453 \tl_put_left:cn { l__enumext_store_save_key_ __enumext_level: _tl }
2454 {
2455 \item \begin{enumext} [
2456 }
2457 \tl_put_right:cn { l__enumext_store_save_key_ __enumext_level: _tl }
2458 {
2459]
2460 }
2461 __enumext_store_addto_seq:v { l__enumext_store_save_key_ __enumext_level: _tl }
2462 }
2463 }
2464 }
2465 \cs_new_protected:Nn __enumext_store_level_close:
2466 {
2467 \bool_if:NT \l__enumext_check_answers_bool
2468 {
2469 __enumext_store_addto_seq:n { \end{enumext} }
2470 }
2471 }

```

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

The “*storing structure*” is handled by the functions `\__enumext_store_level_open_vii:` and `\__enumext_store_level_close_vii:` which are executed in the `enumext*` environment.

```

2472 \cs_new_protected:Nn __enumext_store_level_open_vii:
2473 {
2474 \bool_if:NT \l__enumext_check_answers_bool
2475 {
2476 \tl_if_empty:NTF l__enumext_store_save_key_vii_tl
2477 {
2478 __enumext_store_addto_seq:n
2479 {
2480 \item \begin{enumext*}
2481 }
2482 }
2483 {
2484 \tl_put_left:Nn \l__enumext_store_save_key_vii_tl
2485 {
2486 \item \begin{enumext*}[
2487 }
2488 \tl_put_right:Nn \l__enumext_store_save_key_vii_tl
2489 {
2490]
2491 }
2492 __enumext_store_addto_seq:V l__enumext_store_save_key_vii_tl
2493 }
2494 }
2495 }
2496 \cs_new_protected:Nn __enumext_store_level_close_vii:
2497 {
2498 \bool_if:NT \l__enumext_check_answers_bool
2499 {
2500 __enumext_store_addto_seq:n { \end{enumext*} }
2501 }
2502 }

```

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

### 13.27.7 Function for show marks and position

`\__enumext_print_keyans_box:NN`  
`\__enumext_print_keyans_box:cc`

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

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

```
2503 \cs_new_protected:Nn __enumext_print_keyans_box:NN
2504 {
2505 \mode_leave_vertical:
2506 \skip_horizontal:n { -\dim_use:N #2 }
2507 \makebox[0pt][r]
2508 {
2509 \makebox[\dim_use:N #1][\l__enumext_mark_position_str]
2510 {
2511 \tl_use:N \l__enumext_mark_answer_sym_tl
2512 }
2513 }
2514 \skip_horizontal:n { \dim_use:N #2 }
2515 }
2516 \cs_generate_variant:Nn __enumext_print_keyans_box:NN { cc }
```

(End of definition for `\__enumext_print_keyans_box:NN`.)

### 13.28 The internal label and ref

The function `\__enumext_store_internal_ref:` handles the “*internal label and ref*” system used by the `save-ref` and `mark-ref` keys for `\anskey` will allow to execute `\ref{⟨store name : position⟩}` and will return `1.(a).i.A`.

`\__enumext_store_internal_ref:`

First we will remove the dots “.” from the current `⟨labels⟩`, 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`.

```
2517 \cs_new_protected:Nn __enumext_store_internal_ref:
2518 {
2519 \cs_set_protected:Npn __enumext_tmp:n ##1
2520 {
2521 \tl_set_eq:cc { \l__enumext_label_copy_##1_tl } { \l__enumext_label_##1_tl }
2522 \tl_reverse:c { \l__enumext_label_copy_##1_tl }
2523 \tl_remove_once:cn { \l__enumext_label_copy_##1_tl } { . }
2524 \tl_reverse:c { \l__enumext_label_copy_##1_tl }
2525 }
2526 \clist_map_inline:nn { i, ii, iii, iv, vii } { __enumext_tmp:n {##1} }
2527 \cs_set:Npn __enumext_tmp:n ##1
2528 { . \tl_use:c { \l__enumext_label_copy_ \int_to_roman:n {##1} _tl } }
```

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

```
2529 \bool_lazy_all:nT
2530 {
2531 { \bool_if_p:N \g__enumext_starred_bool }
2532 { \int_compare_p:nNn { \l__enumext_level_int } = { 0 } }
2533 }
2534 {
2535 \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2536 { \tl_use:N \l__enumext_label_copy_vii_tl }
2537 }
2538 \bool_lazy_all:nT
2539 {
2540 { \bool_not_p:n { \g__enumext_standar_bool } }
2541 { \bool_if_p:N \l__enumext_standar_bool }
2542 { \int_compare_p:nNn { \l__enumext_level_int } > { 0 } }
2543 }
2544 {
2545 \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2546 {
2547 \tl_use:N \l__enumext_label_copy_vii_tl
2548 \int_step_function:nnN { 1 } { \l__enumext_level_int } __enumext_tmp:n
2549 }
2550 }
```



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.

```

2551 \bool_lazy_all:nT
2552 {
2553 { \bool_if_p:N \g__enumext_standar_bool }
2554 { \int_compare_p:nNn { \l__enumext_level_int } > { 0 } }
2555 { \int_compare_p:nNn { \l__enumext_level_h_int } = { 0 } }
2556 }
2557 {
2558 \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2559 {
2560 \tl_use:N \l__enumext_label_copy_i_tl
2561 \int_step_function:nnN { 2 } { \l__enumext_level_int } \l__enumext_tmp:n
2562 }
2563 }
2564 \cs_set:Npn \l__enumext_tmp:n ##1
2565 { \tl_use:c { l__enumext_label_copy_ \int_to_roman:n {##1} _tl } . }
2566 \bool_lazy_all:nT
2567 {
2568 { \bool_if_p:N \g__enumext_standar_bool }
2569 { \bool_if_p:N \l__enumext_starred_bool }
2570 { \int_compare_p:nNn { \l__enumext_level_int } > { 0 } }
2571 }
2572 {
2573 \tl_put_right:Ne \l__enumext_newlabel_arg_two_tl
2574 {
2575 \int_step_function:nnN { 1 } { \l__enumext_level_int } \l__enumext_tmp:n
2576 \tl_use:N \l__enumext_label_copy_vii_tl
2577 }
2578 }

```

Now we set the variable `\l__enumext_newlabel_arg_one_tl` which will contain  $\langle \textit{store name} : \textit{position} \rangle$ .

```

2579 \tl_put_right:Ne \l__enumext_newlabel_arg_one_tl
2580 {
2581 \l__enumext_store_name_tl \c_colon_str
2582 \int_eval:n { \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } }
2583 }

```

Now execute the function `\l__enumext_newlabel:nn` and save the result in the variable `\l__enumext_write_aux_file_tl` and finally we write in the `.aux` file.

```

2584 \tl_put_right:Ne \l__enumext_write_aux_file_tl
2585 {
2586 \l__enumext_newlabel:nn
2587 { \exp_not:V \l__enumext_newlabel_arg_one_tl }
2588 { \l__enumext_newlabel_arg_two_tl }
2589 }
2590 \l__enumext_write_aux_file_tl
2591 }

```

(End of definition for `\l__enumext_store_internal_ref:`)

### 13.29 Common functions for `\anskey` and `anskey*` environment

`\l__enumext_store_anskey_code:n`

The internal function `\l__enumext_store_anskey_code:n` first we pass the  $\langle \textit{argument} \rangle$  to the *prop list*, then checks the state of the variable `\l__enumext_store_ref_key_bool` handled by the *save-ref* key and will call the function `\l__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*”  $\langle \textit{argument} \rangle$ .

```

2592 \cs_new_protected:Npn \l__enumext_store_anskey_code:n #1
2593 {
2594 \int_gincr:N \g__enumext_item_anskey_int
2595 \l__enumext_store_addto_prop:n {#1}
2596 \bool_if:NT \l__enumext_store_ref_key_bool
2597 {
2598 \l__enumext_store_internal_ref:
2599 }
2600 \l__enumext_anskey_show_wrap_left:n { #1 }

```

Now we start processing the  $\llbracket \textit{key} = \textit{val} \rrbracket$  passed to the command to build our `\item` in the variable `\l__enumext_store_anskey_arg_tl` which we will “*store*” in the *sequence*. First we clear the variable `\l__enumext_store_anskey_arg_tl` and process the  $\langle \textit{keys} \rangle$ , if the *break-col* key is present and the command is running under `enumext` (not in `enumext*`) we will add `\columnbreak` and then `\item`.

```

2601 \tl_clear:N \l__enumext_store_anskey_arg_tl

```

```

2602 \bool_lazy_and:nnT
2603 { \bool_if_p:N \l__enumext_store_columns_break_bool }
2604 { \bool_not_p:n { \l__enumext_starred_bool } }
2605 {
2606 \tl_put_left:Nn \l__enumext_store_anskey_arg_tl { \columnbreak }
2607 }
2608 \tl_put_right:Nn \l__enumext_store_anskey_arg_tl { \item }

```

If the `item-join` key is present and the command is running under `enumext*` we will add (*number*) to `\l__enumext_store_anskey_arg_tl`.

```

2609 \bool_lazy_and:nnT
2610 { \bool_not_p:n { \l__enumext_starred_bool } }
2611 { \int_compare_p:nNn { \l__enumext_store_item_join_int } > { 1 } }
2612 {
2613 \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2614 {
2615 (\exp_not:V \l__enumext_store_item_join_int)
2616 }
2617 }

```

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*`.

```

2618 \bool_if:NTF \l__enumext_store_item_star_bool
2619 {
2620 \tl_put_right:Nn \l__enumext_store_anskey_arg_tl { * }
2621 \tl_if_empty:NF \l__enumext_store_item_symbol_tl
2622 {
2623 \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2624 {
2625 [\exp_not:V \l__enumext_store_item_symbol_tl]
2626 }
2627 }
2628 \dim_compare:nT
2629 {
2630 \l__enumext_store_item_symbol_sep_dim != \c_zero_dim
2631 }
2632 {
2633 \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2634 {
2635 [\exp_not:V \l__enumext_store_item_symbol_sep_dim]
2636 }
2637 }
2638 \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#1}
2639 }
2640 {
2641 \tl_put_right:Nn \l__enumext_store_anskey_arg_tl {#1}
2642 }

```

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*.

```

2643 \bool_lazy_and:nnT
2644 { \bool_if_p:N \l__enumext_store_ref_key_bool }
2645 { \bool_if_p:N \l__enumext_hyperref_bool }
2646 {
2647 \tl_put_right:Ne \l__enumext_store_anskey_arg_tl
2648 {
2649 \hfill \exp_not:N \hyperlink { \exp_not:V \l__enumext_newlabel_arg_one_tl }
2650 { \exp_not:V \l__enumext_mark_ref_sym_tl }
2651 }
2652 }
2653 __enumext_store_addto_seq:V \l__enumext_store_anskey_arg_tl
2654 }

```

(End of definition for `\__enumext_store_anskey_code:n`)

`\__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-ans` key.

```

2655 \cs_new_protected:Npn __enumext_anskey_show_wrap_arg:n #1
2656 {
2657 \par
2658 \bool_if:NTF \l__enumext_starred_bool
2659 {

```

```

2660 __enumext_print_keyans_box:NN
2661 \l__enumext_labelwidth_vii_dim \l__enumext_labelsep_vii_dim
2662 }
2663 {
2664 __enumext_print_keyans_box:cc
2665 { \l__enumext_labelwidth_ __enumext_level: _dim }
2666 { \l__enumext_labelsep_ __enumext_level: _dim }
2667 }
2668 __enumext_anskey_wrapper:n { #1 }
2669 }

```

(End of definition for \\_\_enumext\_anskey\_show\_wrap\_arg: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  $\langle content \rangle$  stored in the *prop* list 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.

```

2670 \cs_new_protected:Npn __enumext_anskey_show_wrap_left:n #1
2671 {
2672 \bool_if:NT \l__enumext_show_answer_bool
2673 {
2674 __enumext_anskey_show_wrap_arg:n { #1 }
2675 }
2676 \bool_if:NT \l__enumext_show_position_bool
2677 {
2678 \tl_set:Nx \l__enumext_mark_answer_sym_tl
2679 {
2680 \group_begin:
2681 \exp_not:N \normalfont
2682 \exp_not:N \footnotesize [\int_eval:n
2683 {
2684 \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
2685 }
2686]
2687 \group_end:
2688 }
2689 __enumext_anskey_show_wrap_arg:n { #1 }
2690 }
2691 }

```

(End of definition for \\_\_enumext\_anskey\_show\_wrap\_left:n.)

### 13.30 The command \anskey

Since we will be “*storing content*” in a `list` environment within *sequences* 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}`.

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

```

2692 \keys_define:nn { enumext / anskey }
2693 {
2694 break-col .bool_set:N = \l__enumext_store_columns_break_bool,
2695 break-col .default:n = true,
2696 break-col .value_forbidden:n = true,
2697 item-join .int_set:N = \l__enumext_store_item_join_int,
2698 item-join .value_required:n = true,
2699 item-star .bool_set:N = \l__enumext_store_item_star_bool,
2700 item-star .default:n = true,
2701 item-star .value_forbidden:n = true,
2702 item-sym* .tl_set:N = \l__enumext_store_item_symbol_tl,
2703 item-sym* .value_required:n = true,
2704 item-pos* .dim_set:N = \l__enumext_store_item_symbol_sep_dim,
2705 item-pos* .value_required:n = true,
2706 unknown .code:n = { __enumext_anskey_unknown:n {#1} },
2707 }

```

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`.

```

2708 \cs_new_protected:Npn __enumext_anskey_unknown:n #1
2709 {
2710 \exp_args:NV __enumext_anskey_unknown:nn \l_keys_key_str {#1}

```

```

2711 }
2712 \cs_new_protected:Npn __enumext_anskey_unknown:nn #1 #2
2713 {
2714 \tl_if_blank:nTF {#2}
2715 {
2716 \msg_error:nnn { enumext } { anskey-cmd-key-unknown } {#1}
2717 }
2718 {
2719 \msg_error:nnnn { enumext } { anskey-cmd-key-value-unknown } {#1} {#2}
2720 }
2721 }

```

(End of definition for `\__enumext_anskey_unknown:n` and `\__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 `\__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 `[⟨key = val⟩]` and call the function `\__enumext_store_anskey_code:n`.

```

2722 \NewDocumentCommand \anskey { o +m }
2723 {
2724 __enumext_anskey_safe_outer:
2725 \group_begin:
2726 \bool_if:NT __enumext_check_answers_bool
2727 {
2728 \tl_if_no_value:nF {#1}
2729 {
2730 \keys_set:nn { enumext / anskey } {#1}
2731 }
2732 \tl_if_blank:nTF {#2}
2733 {
2734 \msg_error:nn { enumext } { anskey-empty-arg }
2735 }
2736 {
2737 __enumext_anskey_safe_inner:
2738 __enumext_store_anskey_code:n {#2}
2739 }
2740 }
2741 \group_end:
2742 }

```

(End of definition for `\anskey`. This function is documented on page 13.)

### 13.30.1 Internal functions for the command

`\__enumext_anskey_safe_outer:` 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.

`\__enumext_anskey_safe_inner:`

```

2743 \cs_new_protected:Nn __enumext_anskey_safe_outer:
2744 {
2745 \bool_if:NF __enumext_store_active_bool
2746 {
2747 \msg_error:nnnn { enumext } { anskey-wrong-place } { anskey } { enumext }
2748 }
2749 \int_compare:nNt { __enumext_keyans_level_int } = { 1 }
2750 {
2751 \msg_error:nnnn { enumext } { command-wrong-place } { anskey } { keyans }
2752 }
2753 \int_compare:nNt { __enumext_keyans_level_h_int } = { 1 }
2754 {
2755 \msg_error:nnnn { enumext } { command-wrong-place } { anskey } { keyans* }
2756 }
2757 \int_compare:nNt { __enumext_keyans_pic_level_int } = { 1 }
2758 {
2759 \msg_error:nnnn { enumext } { command-wrong-place } { anskey } { keyanspic }
2760 }
2761 }

```

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.

```

2762 \cs_new_protected:Nn __enumext_anskey_safe_inner:
2763 {
2764 \int_incr:N __enumext_anskey_level_int
2765 \int_compare:nNt { __enumext_anskey_level_int } > { 1 }
2766 {
2767 \msg_error:nn { enumext } { anskey-nested }
2768 }
2769 \bool_if:NF __enumext_item_number_bool
2770 {
2771 \msg_error:nn { enumext } { anskey-unnumber-item }
2772 }
2773 \mode_if_math:T
2774 {
2775 \msg_error:nne { enumext } { anskey-math-mode } { \c_backslash_str anskey }
2776 }
2777 }

```

(End of definition for `\__enumext_anskey_safe_outer:` and `\__enumext_anskey_safe_inner:`)

### 13.31 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 *(keys)* 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:` (§13.32) which is executed after the environment in which the key `save-ans` is active.

```

2778 \cs_new_protected:Nn __enumext_undefine_anskey_env:
2779 {
2780 \cs_undefine:c { anskey* }
2781 \cs_undefine:c { endanskey* }
2782 \cs_undefine:c { __scontents_anskey*_env_begin: }
2783 \cs_undefine:c { __scontents_anskey*_env_end: }
2784 }

```

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

```

2785 __enumext_before_env:nn { enumext }
2786 {
2787 \bool_lazy_and:nnT
2788 { \int_compare_p:nNn { __enumext_level_int } = { 0 } }
2789 { \int_compare_p:nNn { __enumext_level_h_int } = { 0 } }
2790 {
2791 \cs_if_free:cF { __scontents_anskey*_env_begin: }
2792 {
2793 \msg_error:nnn { enumext } { anskey-env-error } { anskey* }
2794 }
2795 }
2796 }
2797 __enumext_before_env:nn { enumext* }
2798 {
2799 \bool_lazy_and:nnT
2800 { \int_compare_p:nNn { __enumext_level_int } = { 0 } }
2801 { \int_compare_p:nNn { __enumext_level_h_int } = { 0 } }
2802 {
2803 \cs_if_free:cF { __scontents_anskey*_env_begin: }
2804 {
2805 \msg_error:nnn { enumext } { anskey-env-error } { anskey* }
2806 }
2807 }
2808 }

```

```

2807 }
2808 }

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.

2809 __enumext_before_env:nn { anskey* }
2810 {
2811 \int_compare:nNt { __enumext_keyans_level_int } = { 1 }
2812 {
2813 \msg_error:nnn { enumext } { anskey-env-wrong } { keyans }
2814 }
2815 \int_compare:nNt { __enumext_keyans_level_h_int } = { 1 }
2816 {
2817 \msg_error:nnn { enumext } { anskey-env-wrong } { keyans* }
2818 }
2819 \int_compare:nNt { __enumext_keyans_pic_level_int } = { 1 }
2820 {
2821 \msg_error:nnn { enumext } { anskey-env-wrong } { keyanspic }
2822 }
2823 \bool_if:NF __enumext_item_number_bool
2824 {
2825 \msg_error:nn { enumext } { anskey-unnumber-item }
2826 }
2827 \mode_if_math:T
2828 {
2829 \msg_error:nnn { enumext } { anskey-math-mode } { anskey* }
2830 }
2831 }

```

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

`anskey*`

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:` (§13.26.1) and we will execute it only if the variable `\__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`.

```

2832 \cs_new_protected:Npn __enumext_anskey_env_make:n #1
2833 {
2834 \bool_if:NT __enumext_anskey_env_bool
2835 {
2836 \newenvsc{anskey*}[store-env=#1,print-env=false]
2837 __enumext_anskey_env_exec:
2838 }
2839 }
2840 \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.

2841 \cs_new_protected:Nn __enumext_anskey_env_define_keys:
2842 {
2843 \keys_define:nn { scontents / scontents }
2844 {
2845 break-col .bool_gset:N = \g__enumext_store_columns_break_bool,
2846 break-col .default:n = true,
2847 break-col .value_forbidden:n = true,
2848 item-join .int_gset:N = \g__enumext_store_item_join_int,
2849 item-join .value_required:n = true,
2850 item-star .bool_gset:N = \g__enumext_store_item_star_bool,
2851 item-star .default:n = true,
2852 item-star .value_forbidden:n = true,
2853 item-sym* .tl_gset:N = \g__enumext_store_item_symbol_tl,
2854 item-sym* .value_required:n = true,
2855 item-pos* .dim_gset:N = \g__enumext_store_item_symbol_sep_dim,
2856 item-pos* .value_required:n = true,
2857 print-env .undefine:,
2858 store-env .undefine:,
2859 write-out .undefine:,
2860 unknown .code:n = { __enumext_anskey_env_unknown:n {##1} },

```

```

2861 }
2862 }

```

The *⟨keys⟩* 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`.

```

2863 \cs_new_protected:Npn __enumext_anskey_env_unknown:n #1
2864 {
2865 \exp_args:NV __enumext_anskey_env_unknown:nn \l_keys_key_str {#1}
2866 }
2867 \cs_new_protected:Npn __enumext_anskey_env_unknown:nn #1#2
2868 {
2869 \tl_if_blank:nTF {#2}
2870 {
2871 \msg_error:nnn { enumext } { anskey-env-key-unknown } {#1}
2872 }
2873 {
2874 \msg_error:nnnn { enumext } { anskey-env-key-value-unknown } {#1} {#2}
2875 }
2876 }

```

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`.

```

2877 \cs_new_protected:Nn __enumext_anskey_env_reset_keys:
2878 {
2879 \keys_define:nn { scontents / scontents }
2880 {
2881 break-col .undefine:,
2882 item-join .undefine:,
2883 item-star .undefine:,
2884 item-sym* .undefine:,
2885 item-pos* .undefine:,
2886 write-out .code:n = {
2887 \bool_set_false:N \l_scontents_storing_bool
2888 \bool_set_true:N \l_scontents_writing_bool
2889 \tl_set:Nn \l_scontents_fname_out_tl {##1}
2890 },
2891 write-out .value_required:n = true,
2892 print-env .meta:nn = { scontents } { print-env = ##1 },
2893 print-env .default:n = true,
2894 store-env .meta:nn = { scontents } { store-env = ##1 },
2895 unknown .code:n = { __scontents_parse_environment_keys:n {##1} },
2896 }
2897 }

```

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

```

2898 \cs_new_protected:Npn __enumext_rescan_anskey_env:n #1
2899 {
2900 \group_begin:
2901 \int_set:Nn \tex_newlinechar:D { `^^J }
2902 __scontents_rescan_tokens:x
2903 {
2904 \endgroup % This assumes \catcode`\=0... Things might go off otherwise.
2905 #1
2906 }
2907 }

```

(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 *⟨keys⟩*.

```

2908 \cs_new_protected:Nn __enumext_anskey_env_exec:
2909 {
2910 __enumext_before_env:nn { anskey* }
2911 {
2912 __enumext_anskey_env_define_keys:
2913 }

```



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.

```

2914 \hook_if_empty:nF {env/anskey*/after}
2915 {
2916 \hook_gremove_code:nn {env/anskey*/after} { * }
2917 }
2918 __enumext_after_env:nn { anskey* }
2919 {
2920 __enumext_anskey_env_save_keys:
2921 \tl_clear:N \l__enumext_store_anskey_env_tl
2922 \tl_clear:N \l__enumext_store_anskey_opt_tl
2923 \bool_if:NT \l__enumext_check_answers_bool
2924 {
2925 \tl_gset:Ne \l__enumext_store_anskey_env_tl
2926 {
2927 \seq_item:ce { g__scontents_name_ \l__enumext_store_name_tl _seq } { -1 }
2928 }
2929 \regex_match:nVTF
2930 { ^\s* \z | ^\s* \u{c__scontents_hidden_space_str} \z }
2931 \l__enumext_store_anskey_env_tl
2932 {
2933 \msg_error:nn { enumext } { anskey-empty-arg }
2934 }
2935 {
2936 __enumext_anskey_env_store:
2937 }
2938 }
2939 __enumext_anskey_env_clean_vars:
2940 __enumext_anskey_env_reset_keys:
2941 }
2942 }

```

The use of `\hook_gremove_code:nn` is necessary here, otherwise the `{⟨code⟩}` 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 `[⟨key = val⟩]` passed to the environment and save this in the variable `\l__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`.

```

2943 \cs_new_protected:Nn __enumext_anskey_env_save_keys:
2944 {
2945 \bool_lazy_and:nnT
2946 { \bool_if_p:N \g__enumext_store_columns_break_bool }
2947 { \bool_not_p:n { \l__enumext_starred_bool } }
2948 {
2949 \tl_put_left:Ne \l__enumext_store_anskey_opt_tl { ,break-col, }
2950 }

```

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

```

2951 \bool_lazy_and:nnT
2952 { \bool_not_p:n { \l__enumext_starred_bool } }
2953 { \int_compare_p:nNn { \g__enumext_store_item_join_int } > { 1 } }
2954 {
2955 \tl_put_left::Ne \l__enumext_store_anskey_opt_tl
2956 {
2957 ,item-join = \exp_not:V \g__enumext_store_item_join_int,
2958 }
2959 }

```

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

```

2960 \bool_if:NT \g__enumext_store_item_star_bool
2961 {
2962 \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
2963 {
2964 ,item-star,
2965 }
2966 \tl_if_empty:NF \g__enumext_store_item_symbol_tl

```

```

2967 {
2968 \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
2969 {
2970 ,item-sym* = \exp_not:V \g__enumext_store_item_symbol_tl,
2971 }
2972 }
2973 \dim_compare:nT
2974 {
2975 \g__enumext_store_item_symbol_sep_dim != \c_zero_dim
2976 }
2977 {
2978 \tl_put_left:Ne \l__enumext_store_anskey_opt_tl
2979 {
2980 ,item-pos* = \exp_not:V \g__enumext_store_item_symbol_sep_dim,
2981 }
2982 }
2983 }
2984 }

```

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`.

```

2985 \cs_new_protected:Nn __enumext_anskey_env_store:
2986 {
2987 \group_begin:
2988 \tl_if_empty:NTF \l__enumext_store_anskey_opt_tl
2989 {
2990 \exp_args:Ne
2991 __enumext_store_anskey_code:n
2992 {
2993 __enumext_rescan_anskey_env:n { \l__enumext_store_anskey_env_tl }
2994 }
2995 }
2996 {
2997 \keys_set_known:nV { enumext / anskey } \l__enumext_store_anskey_opt_tl
2998 \exp_args:Ne
2999 __enumext_store_anskey_code:n
3000 {
3001 __enumext_rescan_anskey_env:n { \l__enumext_store_anskey_env_tl }
3002 }
3003 }
3004 \group_end:
3005 }

```

The function `\__enumext_anskey_env_clean_vars:` will return the global variables used by the `<keys>` to their initial state.

```

3006 \cs_new_protected:Nn __enumext_anskey_env_clean_vars:
3007 {
3008 \bool_gset_false:N \g__enumext_store_columns_break_bool
3009 \int_gzero:N \g__enumext_store_item_join_int
3010 \bool_gset_false:N \g__enumext_store_item_star_bool
3011 \tl_gclear:N \g__enumext_store_item_symbol_tl
3012 \dim_gzero:N \g__enumext_store_item_symbol_sep_dim
3013 }

```

(End of definition for `\__enumext_anskey_env_save_keys:`, `\__enumext_anskey_env_store:`, and `\__enumext_anskey_env_clean_vars:`.)

### 13.32 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*` (§13.31) 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.

```

3014 \cs_new_protected:Nn __enumext_execute_after_env:
3015 {
3016 \int_compare:nNnT { \l__enumext_level_int } = { 0 }
3017 {
3018 \tl_if_empty:NF \g__enumext_store_name_tl

```

```

3019 {
3020 __enumext_stop_save_ans_msg:
3021 __enumext_item_answer_diff:
3022 __enumext_log_global_vars:
3023 __enumext_log_answer_vars:
3024 \bool_if:NTF \g__enumext_check_ans_key_bool
3025 {
3026 __enumext_check_ans_show:
3027 }
3028 { __enumext_check_ans_log: }
3029 __enumext_undefine_anskey_env:
3030 }
3031 __enumext_reset_global_vars:
3032 }
3033 }

```

(End of definition for \\_\_enumext\_execute\_after\_env:.)

- This function is passed to the function \\_\_enumext\_after\_env:nn for the environments `enumext` (§13.39) and `enumext*` (§13.44) and it is executed only when the environments are not nested or at some level of these..

### 13.33 Common functions for keyans, keyans\* and keyanspic

#### 13.33.1 Storing content in prop list

\\_\_enumext\_keyans\_addto\_prop:n

The function \\_\_enumext\_keyans\_addto\_prop:n will pass the the current *⟨label⟩* for *\item\** in *keyans* environment and the current *⟨label⟩* for *\anspic\** in *keyanspic* environment followed by the *⟨contents⟩* of the *optional argument* of both commands to the \\_\_enumext\_store\_current\_label\_tl variable, which will be stored to the *prop list* defined by the *save-ans* key using the function \\_\_enumext\_store\_addto\_prop:V.

```

3034 \cs_new_protected:Npn __enumext_keyans_addto_prop:n #1
3035 {
3036 \tl_clear:N __enumext_store_current_label_tl
3037 \int_compare:nNnTF { __enumext_keyans_pic_level_int } = { 1 }
3038 {
3039 \tl_put_right:Ne __enumext_store_current_label_tl { __enumext_label_vi_tl }
3040 }
3041 {
3042 \tl_put_right:Ne __enumext_store_current_label_tl { __enumext_label_v_tl }
3043 }

```

If the *optional argument* is present and the *save-sep* key is not empty, we save it.

```

3044 \tl_if_novalue:nF { #1 }
3045 {
3046 \tl_if_empty:NF __enumext_store_keyans_item_opt_sep_tl
3047 {
3048 \tl_put_right:Ne __enumext_store_current_label_tl
3049 {
3050 __enumext_store_keyans_item_opt_sep_tl
3051 }
3052 }
3053 \tl_put_right:Ne __enumext_store_current_label_tl { #1 }
3054 }
3055 __enumext_store_addto_prop:V __enumext_store_current_label_tl
3056 }

```

(End of definition for \\_\_enumext\_keyans\_addto\_prop:n.)

#### 13.33.2 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 *\anskey* basically because in this environments the interest is in the current *⟨label⟩* for *\item\** and *\anspic\** with the *⟨contents⟩* of the *optional argument*. The mechanism defined here will allow to execute *\ref{⟨store name : position⟩}* and will return 1. (A).

\\_\_enumext\_keyans\_store\_ref:  
 \\_\_enumext\_keyans\_store\_ref\_aux\_i:  
 \\_\_enumext\_keyans\_store\_ref\_aux\_ii:

The function \\_\_enumext\_keyans\_store\_ref: handles the “*internal label and ref*” system used by the *save-ref* key for *\item\** and *\anspic\** commands. First we will create copies of the current *⟨labels⟩* and remove the dots “.” from them, we do not want to get double dots in references.

```

3057 \cs_new_protected:Nn __enumext_keyans_store_ref:
3058 {
3059 \bool_if:NT __enumext_store_ref_key_bool
3060 {
3061 \cs_set_protected:Npn __enumext_tmp:n ##1
3062 {

```

```

3063 \tl_set_eq:cc { __enumext_label_copy_##1_tl } { __enumext_label_##1_tl }
3064 \tl_reverse:c { __enumext_label_copy_##1_tl }
3065 \tl_remove_once:cn { __enumext_label_copy_##1_tl } { . }
3066 \tl_reverse:c { __enumext_label_copy_##1_tl }
3067 }
3068 \clist_map_inline:nn { i, v, vi, vii, viii } { __enumext_tmp:n {##1} }
3069 __enumext_keyans_store_ref_aux_i:
3070 }
3071 }

```

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

```

3072 \cs_new_protected:Nn __enumext_keyans_store_ref_aux_i:
3073 {
3074 \bool_if:NT \g__enumext_starred_bool
3075 {
3076 \tl_set_eq:NN __enumext_label_copy_i_tl __enumext_label_copy_vii_tl
3077 }
3078 \int_compare:nNnT { __enumext_keyans_pic_level_int } = { 1 }
3079 {
3080 \tl_put_right:Ne __enumext_newlabel_arg_two_tl
3081 { __enumext_label_copy_i_tl . __enumext_label_copy_vi_tl }
3082 }
3083 \int_compare:nNnT { __enumext_keyans_level_int } = { 1 }
3084 {
3085 \tl_put_right:Ne __enumext_newlabel_arg_two_tl
3086 { __enumext_label_copy_i_tl . __enumext_label_copy_v_tl }
3087 }
3088 \int_compare:nNnT { __enumext_keyans_level_h_int } = { 1 }
3089 {
3090 \tl_put_right:Ne __enumext_newlabel_arg_two_tl
3091 { __enumext_label_copy_i_tl . __enumext_label_copy_viii_tl }
3092 }
3093 \tl_put_right:Ne __enumext_newlabel_arg_one_tl
3094 {
3095 __enumext_store_name_tl \c_colon_str
3096 \int_eval:n { \prop_count:c { g__enumext_ __enumext_store_name_tl _prop } }
3097 }
3098 __enumext_keyans_store_ref_aux_ii:
3099 }

```

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.

```

3100 \cs_new_protected:Nn __enumext_keyans_store_ref_aux_ii:
3101 {
3102 \tl_put_right:Ne __enumext_write_aux_file_tl
3103 {
3104 __enumext_newlabel:nn
3105 { \exp_not:V __enumext_newlabel_arg_one_tl }
3106 { __enumext_newlabel_arg_two_tl }
3107 }
3108 __enumext_write_aux_file_tl
3109 }

```

(End of definition for `\__enumext_keyans_store_ref:`, `\__enumext_keyans_store_ref_aux_i:`, and `\__enumext_keyans_store_ref_aux_ii:`.)

### 13.33.3 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  $\langle \textit{label} \rangle$  `\__enumext_label_v_tl` for the `keyans` environment and the `\__enumext_label_vi_tl` for the `keyanspic` environment when using `\item*` and `\anspic*`, followed by the  $\langle \textit{contents} \rangle$  of the *optional argument* of both commands to the `\__enumext_store_current_label_tl` variable to the sequence defined by the `save-ans` key.

```

3110 \cs_new_protected:Npn __enumext_keyans_addto_seq:n #1
3111 {
3112 \tl_clear:N __enumext_store_current_label_tl
3113 \int_compare:nNnTF { __enumext_keyans_pic_level_int } = { 1 }
3114 {
3115 \tl_put_right:Ne __enumext_store_current_label_tl { \item __enumext_label_vi_tl }
3116 }

```

```

3117 {
3118 \tl_put_right:Ne \l__enumext_store_current_label_tl { \item \l__enumext_label_v_tl }
3119 }
3120 \tl_if_novalue:nF { #1 }
3121 {
3122 \tl_if_empty:NF \l__enumext_store_keyans_item_opt_sep_tl
3123 {
3124 \tl_put_right:Ne \l__enumext_store_current_label_tl
3125 {
3126 \l__enumext_store_keyans_item_opt_sep_tl
3127 }
3128 }
3129 \tl_put_right:Ne \l__enumext_store_current_label_tl { #1 }
3130 }
3131 __enumext_keyans_addto_seq_link:
3132 }

```

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 `check-ans` key.

```

3133 \cs_new_protected:Nn __enumext_keyans_addto_seq_link:
3134 {
3135 \bool_lazy_and:nnT
3136 { \bool_if_p:N \l__enumext_store_ref_key_bool }
3137 { \bool_if_p:N \l__enumext_hyperref_bool }
3138 {
3139 \tl_put_right:Ne \l__enumext_store_current_label_tl
3140 {
3141 \hfill \exp_not:N \hyperlink
3142 {
3143 \exp_not:V \l__enumext_newlabel_arg_one_tl
3144 }
3145 { \exp_not:V \l__enumext_mark_ref_sym_tl }
3146 }
3147 }
3148 __enumext_store_addto_seq:V \l__enumext_store_current_label_tl
3149 \bool_if:NT \l__enumext_check_answers_bool
3150 {
3151 \int_gincr:N \g__enumext_item_anskey_int
3152 }
3153 }

```

(End of definition for `\__enumext_keyans_addto_seq:n` and `\__enumext_keyans_addto_seq_link:.`)

### 13.33.4 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 `(label)` are incorrect.

```

__enumext_keyans_show_left:n
__enumext_keyans_show_ans:
__enumext_keyans_show_pos:
__enumext_keyans_show_item_opt:

```

Common function to show *starred commands* `\item*` and `(position)` of stored content in *prop list* for `keyans` and `keyanspic`. Need add `1` to `\g__enumext_⟨store name⟩_prop` for `show-pos` key.

```

3154 \cs_new_protected:Npn __enumext_keyans_show_left:n #1
3155 {
3156 \tl_if_novalue:nF { #1 }
3157 {
3158 \tl_set:Ne \l__enumext_store_current_opt_arg_tl { #1 }
3159 }
3160 \bool_if:NT \l__enumext_show_answer_bool
3161 {
3162 __enumext_keyans_show_ans:
3163 }
3164 \bool_if:NT \l__enumext_show_position_bool
3165 {
3166 __enumext_keyans_show_pos:
3167 }
3168 }
3169 \cs_new_protected:Nn __enumext_keyans_show_item_opt:
3170 {
3171 \tl_if_empty:NF \l__enumext_store_current_opt_arg_tl

```

```

3172 {
3173 \bool_lazy_or:nnT
3174 { \bool_if_p:N \l__enumext_show_answer_bool }
3175 { \bool_if_p:N \l__enumext_show_position_bool }
3176 {
3177 __enumext_keyans_wrapper_opt:n { \l__enumext_store_current_opt_arg_tl } \c_space_tl
3178 }
3179 }
3180 }
3181 \cs_new_protected:Nn __enumext_keyans_show_ans:
3182 {
3183 \bool_if:NT \l__enumext_starred_bool
3184 {
3185 \dim_set_eq:NN \l__enumext_labelwidth_i_dim \l__enumext_labelwidth_vii_dim
3186 \dim_set_eq:NN \l__enumext_labelsep_i_dim \l__enumext_labelsep_vii_dim
3187 }
3188 \tl_put_left:Nn \l__enumext_label_v_tl
3189 {
3190 __enumext_print_keyans_box:NN
3191 \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
3192 }
3193 }
3194 \cs_new_protected:Nn __enumext_keyans_show_pos:
3195 {
3196 \bool_if:NT \l__enumext_starred_bool
3197 {
3198 \dim_set_eq:NN \l__enumext_labelwidth_i_dim \l__enumext_labelwidth_vii_dim
3199 \dim_set_eq:NN \l__enumext_labelsep_i_dim \l__enumext_labelsep_vii_dim
3200 }
3201 \int_compare:nNnTF { \l__enumext_keyans_pic_level_int } = { 1 }
3202 {
3203 \tl_set:Ne \l__enumext_mark_answer_sym_tl
3204 {
3205 \group_begin:
3206 \exp_not:N \normalfont
3207 \exp_not:N \footnotesize [\int_eval:n
3208 {
3209 \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
3210 }
3211]
3212 \group_end:
3213 }
3214 }
3215 {
3216 \tl_set:Ne \l__enumext_mark_answer_sym_tl
3217 {
3218 \group_begin:
3219 \exp_not:N \normalfont
3220 \exp_not:N \footnotesize [\int_eval:n
3221 {
3222 \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop } + 1
3223 }
3224]
3225 \group_end:
3226 }
3227 }
3228 \tl_put_left:Nn \l__enumext_label_v_tl
3229 {
3230 __enumext_print_keyans_box:NN
3231 \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
3232 }
3233 }

```

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

### 13.34 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.

When *labeling* PDF is active `\makeLabel` is redefined as `\hss #1` and the only way to get the `align` key to work correctly is to redefine `\makeLabel` using `\makebox`. The best way to implement this is to use the

conditional command `\IfDocumentMetadataTF` to force this redefinition and the dedicated `mode-box` key to manually activate it by the user.

The `\item` and `\item[⟨custom⟩]` commands work in the usual way on `enumext` and we will add `\item*`, `\item*[⟨symbol⟩]` and `\item*[⟨symbol⟩][⟨offset⟩]`.

`\__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`.

```

3234 \cs_new_protected:Npn __enumext_default_item:n #1
3235 {
3236 \tl_if_novalue:nTF {#1}
3237 {
3238 \bool_if:NT \l__enumext_check_answers_bool
3239 {
3240 \int_gincr:N \g__enumext_item_number_int
3241 \bool_set_true:N \l__enumext_item_number_bool
3242 }
3243 \bool_set_true:c { l__enumext_wrap_label_ __enumext_level: _bool }
3244 __enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ __enumext_level: _tl }
3245 }
3246 {
3247 \bool_set_eq:cc
3248 { l__enumext_wrap_label_ __enumext_level: _bool }
3249 { l__enumext_wrap_label_opt_ __enumext_level: _bool }
3250 __enumext_item_std:w [#1] \tl_use:c { l__enumext_fake_item_indent_ __enumext_level: _tl }
3251 }
3252 }

```

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

`\__enumext_starred_item:nn`

`\__enumext_item_star_exec:`

The `\item*`, `\item*[⟨symbol⟩]` and `\item*[⟨symbol⟩][⟨offset⟩]` works like the *numbered* `\item`, but placing a `⟨symbol⟩` to the “left” of the `⟨label⟩` separated from it by the value the second *optional argument* `⟨offset⟩`.

**#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`.

```

3253 \cs_new_protected:Npn __enumext_starred_item:nn #1 #2
3254 {
3255 \tl_if_novalue:nTF {#1}
3256 {
3257 \tl_gset_eq:Nc
3258 \g__enumext_item_symbol_aux_tl { l__enumext_item_symbol_ __enumext_level: _tl }
3259 }
3260 {
3261 \tl_gset:Nn \g__enumext_item_symbol_aux_tl {#1}
3262 }
3263 \tl_if_novalue:nTF {#2}
3264 {
3265 \dim_set_eq:cc
3266 { l__enumext_item_symbol_sep_ __enumext_level: _dim }
3267 { l__enumext_labelsep_ __enumext_level: _dim }
3268 }
3269 {
3270 \dim_set:cn { l__enumext_item_symbol_sep_ __enumext_level: _dim } {#2}
3271 }
3272 \bool_if:NT \l__enumext_check_answers_bool
3273 {
3274 \int_gincr:N \g__enumext_item_number_int
3275 \bool_set_true:N \l__enumext_item_number_bool
3276 }
3277 \bool_set_true:c { l__enumext_wrap_label_ __enumext_level: _bool }

```



```

3278 __enumext_item_std:w \tl_use:c { l__enumext_fake_item_indent_ __enumext_level: _tl }
3279 }

```

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

```

3280 \cs_new_protected:Nn __enumext_item_star_exec:
3281 {
3282 \tl_if_empty:cF { l__enumext_item_symbol_ __enumext_level: _tl }
3283 {
3284 \mode_leave_vertical:
3285 \skip_horizontal:n { -\dim_use:c { l__enumext_item_symbol_sep_ __enumext_level: _dim } }
3286 \hbox_overlap_left:n { \g__enumext_item_symbol_aux_tl }
3287 \skip_horizontal:n { \dim_use:c { l__enumext_item_symbol_sep_ __enumext_level: _dim } }
3288 }
3289 }

```

(End of definition for `\__enumext_starred_item:nn` and `\__enumext_item_star_exec:`.)

`\__enumext_redefine_item:` The function `\__enumext_redefine_item:` will redefine the `\item` command in the `enumext` environment adding `\item*`. This function are passed to `\__enumext_list_arg_two_X:` used in the definition of the `enumext` environment (§13.39).

```

3290 \cs_new_protected:Nn __enumext_redefine_item:
3291 {
3292 \RenewDocumentCommand \item { s o o }
3293 {
3294 \bool_if:nTF {##1}
3295 {
3296 __enumext_starred_item:nn {##2} {##3}
3297 }
3298 { __enumext_default_item:n {##2} }
3299 }
3300 }

```

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

`\__enumext_make_label:` The function `\__enumext_make_label:` redefine `\makelabel` for the keys `mode-box`, `align`, `font`, `wrap-label`, `wrap-label*` and `\item*` for `enumext` environment. This function are passed to `\__enumext_list_arg_two_X:` used in the definition of the `enumext` environment (§13.39).

```

3301 \cs_new_protected:Nn __enumext_make_label:
3302 {
3303 \IfDocumentMetadataTF
3304 {
3305 __enumext_make_label_box:
3306 }
3307 {
3308 \bool_if:NTF \l__enumext_mode_box_bool
3309 {
3310 __enumext_make_label_box:
3311 }
3312 {
3313 __enumext_make_label_std:
3314 }
3315 }
3316 }

```

Standard definition when `\DocumentMetadata` is not active.

```

3317 \cs_new_protected:Nn __enumext_make_label_std:
3318 {
3319 \RenewDocumentCommand \makelabel { m }
3320 {
3321 \tl_use:c { l__enumext_label_fill_left_ __enumext_level: _tl }
3322 \tl_use:c { l__enumext_label_font_style_ __enumext_level: _tl }
3323 \bool_if:cTF { l__enumext_wrap_label_ __enumext_level: _bool }
3324 {
3325 __enumext_item_star_exec:
3326 \use:c { __enumext_wrapper_label_ __enumext_level: :n } { ##1 }
3327 }
3328 { ##1 }
3329 \tl_use:c { l__enumext_label_fill_right_ __enumext_level: _tl }
3330 \tl_gclear:N \g__enumext_item_symbol_aux_tl
3331 }
3332 }

```

Definition using `\makebox` when `\DocumentMetadata` is active or `mode-box` is active.

- Here it is necessary to use `\strut\smash` to maintain text *alignment* in case the user wants to use `\labelbx` for example. In my experiments with *mimicking* the `description` environment it was the only way out and it seems to have no adverse effects and may serve in the future as a basis for a more generic `list` environment package than `enumext`.

```

3333 \cs_new_protected:Nn __enumext_make_label_box:
3334 {
3335 \RenewDocumentCommand \make_label { m }
3336 {
3337 \strut\smash
3338 {
3339 \makebox
3340 [\dim_use:c { l__enumext_labelwidth_ __enumext_level: _dim }]
3341 [\str_use:c { l__enumext_align_label_pos_ __enumext_level: _str }]
3342 {
3343 \tl_use:c { l__enumext_label_font_style_ __enumext_level: _tl }
3344 \bool_if:cTF { l__enumext_wrap_label_ __enumext_level: _bool }
3345 {
3346 __enumext_item_star_exec:
3347 \use:c { __enumext_wrapper_label_ __enumext_level: :n } { ##1 }
3348 }
3349 { ##1 }
3350 \tl_gclear:N \g__enumext_item_symbol_aux_tl
3351 }
3352 } % close smash
3353 }
3354 }

```

(End of definition for `\__enumext_make_label:`, `\__enumext_make_label_std:`, and `\__enumext_make_label_box:`.)

### 13.35 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 `<offset>`.

`item-sym*` Define and set `item-sym*` and `item-pos*` keys for `enumext` and `enumext*`.

```

item-pos*
3355 \cs_set_protected:Npn __enumext_tmp:nn #1 #2
3356 {
3357 \keys_define:nn { enumext / #1 }
3358 {
3359 item-sym* .tl_set:c = { l__enumext_item_symbol_#2_tl },
3360 item-sym* .value_required:n = true,
3361 item-sym* .initial:n = {\textasteriskcentered},
3362 item-pos* .dim_set:c = { l__enumext_item_symbol_sep_#2_dim },
3363 item-pos* .value_required:n = true,
3364 }
3365 }
3366 \clist_map_inline:nn
3367 {
3368 {level-1}{i}, {level-2}{ii}, {level-3}{iii}, {level-4}{iv}, {enumext*}{vii}
3369 }
3370 { __enumext_tmp:nn #1 }

```

(End of definition for `item-sym*` and `item-pos*`.)

### 13.36 Handling unknown keys

At this point in the code I already know that I will not add more `<keys>` 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 `<keys>` (you have to be consistent in life).

#### 13.36.1 Handling unknown keys for `keyans`, `keyans*` and `keyanspic`

`unknown` Define and set `unknown` key for `keyans`, `keyans*` and `keyanspic` environments.

```

__enumext_keyans_unknown_keys:n
__enumext_keyans_unknown_keys:nn
3371 \cs_set_protected:Npn __enumext_tmp:n #1
3372 {
3373 \keys_define:nn { enumext / #1 }
3374 {
3375 unknown .code:n = { __enumext_keyans_unknown_keys:n {##1} }
3376 }
3377 }
3378 \clist_map_inline:nn { keyans, keyans*, keyanspic } { __enumext_tmp:n {#1} }

```

Internal functions for handling `unknown` key.

```

3379 \cs_new_protected:Npn __enumext_keyans_unknown_keys:n #1
3380 {
3381 \exp_args:NV __enumext_keyans_unknown_keys:nn \l_keys_key_str {#1}
3382 }
3383 \cs_new_protected:Npn __enumext_keyans_unknown_keys:nn #1#2
3384 {
3385 \tl_if_blank:nTF {#2}
3386 {
3387 \msg_error:nnn { enumext } { keyans-unknown-key } {#1}
3388 }
3389 {
3390 \msg_error:nnnn { enumext } { keyans-unknown-key-value } {#1} {#2}
3391 }
3392 }

```

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

### 13.36.2 Handling unknown keys for `enumext*`

`unknown`

Define and set `unknown` key for `enumext*` environment.

```

__enumext_starred_unknown_keys:n
__enumext_starred_unknown_keys:nn
3393 \keys_define:nn { enumext / enumext* }
3394 {
3395 unknown .code:n = { __enumext_starred_unknown_keys:n {#1} }
3396 }

```

Internal functions for handling `unknown` key.

```

3397 \cs_new_protected:Npn __enumext_starred_unknown_keys:n #1
3398 {
3399 \exp_args:NV __enumext_starred_unknown_keys:nn \l_keys_key_str {#1}
3400 }
3401 \cs_new_protected:Npn __enumext_starred_unknown_keys:nn #1#2
3402 {
3403 \tl_if_blank:nTF {#2}
3404 {
3405 \msg_error:nnn { enumext } { starred-unknown-key } {#1}
3406 }
3407 {
3408 \msg_error:nnnn { enumext } { starred-unknown-key-value } {#1} {#2}
3409 }
3410 }

```

(End of definition for `unknown`, `\__enumext_starred_unknown_keys:n`, and `\__enumext_starred_unknown_keys:nn`.)

### 13.36.3 Handling unknown keys for `enumext`

`unknown`

Defines and set the key `unknown` for `enumext` environment.

```

__enumext_standar_unknown_keys:n
__enumext_standar_unknown_keys:nn
3411 \cs_set_protected:Npn __enumext_tmp:n #1
3412 {
3413 \keys_define:nn { enumext / #1 }
3414 {
3415 unknown .code:n = { __enumext_standar_unknown_keys:n {##1} }
3416 }
3417 }
3418 \clist_map_inline:nn { level-1,level-2,level-3,level-4 } { __enumext_tmp:n {#1} }

```

Internal functions for handling `unknown` key.

```

3419 \cs_new_protected:Npn __enumext_standar_unknown_keys:n #1
3420 {
3421 \exp_args:NV __enumext_standar_unknown_keys:nn \l_keys_key_str {#1}
3422 }
3423 \cs_new_protected:Npn __enumext_standar_unknown_keys:nn #1#2
3424 {
3425 \tl_if_blank:nTF {#2}
3426 {
3427 \msg_error:nnn { enumext } { standar-unknown-key } {#1}
3428 }
3429 {
3430 \msg_error:nnnn { enumext } { standar-unknown-key-value } {#1} {#2}
3431 }
3432 }

```

(End of definition for `unknown`, `\__enumext_standar_unknown_keys:n`, and `\__enumext_standar_unknown_keys:nn`.)

### 13.37 Redefining \item and \makeLabel in keyans

The `\item` and `\item[⟨custom⟩]` commands work in the usual way in `keyans`, but the `\item*` and `\item*[⟨content⟩]` commands *store* the current `⟨label⟩` next to the `⟨content⟩` if it is present in the *sequence* and *prop list* defined by `save-ans` key.

`\__enumext_keyans_default_item:n`

The function `\__enumext_keyans_default_item:n` executes the original behavior of the `\item` along with the keys `wrap-label`, `wrap-label*` and `itemindent`.

```

3433 \cs_new_protected:Npn __enumext_keyans_default_item:n #1
3434 {
3435 \tl_if_novalue:nTF { #1 }
3436 {
3437 \bool_set_true:N __enumext_wrap_label_v_bool
3438 __enumext_item_std:w \tl_use:N __enumext_fake_item_indent_v_tl
3439 }
3440 {
3441 \bool_set_eq:NN __enumext_wrap_label_v_bool __enumext_wrap_label_opt_v_bool
3442 __enumext_item_std:w [#1] \tl_use:N __enumext_fake_item_indent_v_tl
3443 }
3444 }

```

(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 `⟨label⟩`, 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 `⟨label⟩`, followed by this it will execute function `\__enumext_keyans_show_item_opt:` handled by `wrap-opt` key.

```

3445 \cs_new_protected:Npn __enumext_keyans_starred_item:n #1
3446 {
3447 \tl_set_eq:NN __enumext_store_current_label_tmp_tl __enumext_label_v_tl
3448 __enumext_keyans_show_left:n { #1 }
3449 \bool_set_true:N __enumext_wrap_label_v_bool
3450 __enumext_item_std:w \tl_use:N __enumext_fake_item_indent_v_tl
3451 __enumext_keyans_show_item_opt:

```

Recover the original value of the current `⟨label⟩` and *store* it first in the *prop list* (including the *optional argument*), run the internal “*label and ref*” system if the `save-ref` key is active, *store* it in the *sequence* and finally increments `\g__enumext_check_starred_cmd_int` for internal check system.

```

3452 \tl_set_eq:NN __enumext_label_v_tl __enumext_store_current_label_tmp_tl
3453 __enumext_keyans_addto_prop:n { #1 }
3454 __enumext_keyans_store_ref:
3455 __enumext_keyans_addto_seq:n { #1 }
3456 \int_gincr:N \g__enumext_check_starred_cmd_int
3457 }

```

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

`\item*`

`\__enumext_keyans_redefine_item:`

The function `\__enumext_keyans_redefine_item:` is responsible for adding the *starred argument* and *optional argument* by the `\__enumext_list_arg_two_v:` function in the definition of the `keyans` environment. Here we need to use `\peek_remove_spaces:n` to prevent an unwanted space when using `\item*` in conjunction with the `itemindent` key. This function are passed to `\__enumext_list_arg_two_v:` used in the definition of the `keyans` environment (§13.38).

```

3458 \cs_new_protected:Nn __enumext_keyans_redefine_item:
3459 {
3460 \RenewDocumentCommand \item { s o }
3461 {
3462 \bool_if:nTF {##1}
3463 {
3464 \int_incr:N __enumext_item_star_exec_int % increment
3465 \peek_remove_spaces:n
3466 {
3467 __enumext_keyans_starred_item:n {##2}
3468 }
3469 }
3470 {
3471 \int_zero:N __enumext_item_star_exec_int % zero
3472 __enumext_keyans_default_item:n {##2}
3473 }
3474 }
3475 }

```

(End of definition for `\item*` and `\__enumext_keyans_redefine_item:`. This function is documented on page 15.)

```
__enumext_keyans_make_label:
__enumext_keyans_wrapper_label:n
__enumext_keyans_make_label_std:
__enumext_keyans_make_label_box:
```

The function `\__enumext_keyans_make_label:` redefine `\makeLabel` for the keys `mode-box`, `align`, `font`, `wrap-label`, `wrap-label*` and `\item*` for `keyans` environment. This function are passed to `\__enumext_list_arg_two_v:` used in the definition of the `keyans` environment (§13.38).

```
3476 \cs_new_protected:Nn __enumext_keyans_make_label:
3477 {
3478 \IfDocumentMetadataTF
3479 {
3480 __enumext_keyans_make_label_box:
3481 }
3482 {
3483 \bool_if:NTF \l__enumext_mode_box_bool
3484 {
3485 __enumext_keyans_make_label_box:
3486 }
3487 {
3488 __enumext_keyans_make_label_std:
3489 }
3490 }
3491 }
```

Standard definition when `\DocumentMetadata` is not active.

```
3492 \cs_new_protected:Npn __enumext_keyans_wrapper_label:n #1
3493 {
3494 \bool_lazy_all:nT
3495 {
3496 { \bool_if_p:N \l__enumext_wrap_label_v_bool }
3497 { \bool_if_p:N \l__enumext_show_answer_bool }
3498 { \int_compare_p:nNn { \l__enumext_item_star_exec_int } = { 1 } }
3499 { \cs_if_exist_p:N __enumext_keyans_wrapper_item:n }
3500 }
3501 {
3502 \cs_set_eq:NN __enumext_wrapper_label_v:n __enumext_keyans_wrapper_item:n
3503 }
3504 \bool_if:NTF \l__enumext_wrap_label_v_bool
3505 {
3506 __enumext_wrapper_label_v:n { #1 }
3507 }
3508 { #1 }
3509 }
3510 \cs_new_protected:Nn __enumext_keyans_make_label_std:
3511 {
3512 \RenewDocumentCommand \makeLabel { m }
3513 {
3514 \tl_use:N \l__enumext_label_fill_left_v_tl
3515 \tl_use:N \l__enumext_label_font_style_v_tl
3516 __enumext_keyans_wrapper_label:n { ##1 }
3517 \tl_use:N \l__enumext_label_fill_right_v_tl
3518 }
3519 }
```

Definition using `\makebox` when `\DocumentMetadata` is active or `mode-box` is active.

```
3520 \cs_new_protected:Nn __enumext_keyans_make_label_box:
3521 {
3522 \RenewDocumentCommand \makeLabel { m }
3523 {
3524 \strut\smash
3525 {
3526 \makebox[\l__enumext_labelwidth_v_dim][\l__enumext_align_label_pos_v_str]
3527 {
3528 \tl_use:N \l__enumext_label_font_style_v_tl
3529 __enumext_keyans_wrapper_label:n { ##1 }
3530 }
3531 }
3532 }
3533 }
```

(End of definition for `\__enumext_keyans_make_label:` and others.)

### 13.38 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`.

#### 13.38.1 Calculation of `\leftmargin` and `\itemindent`

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

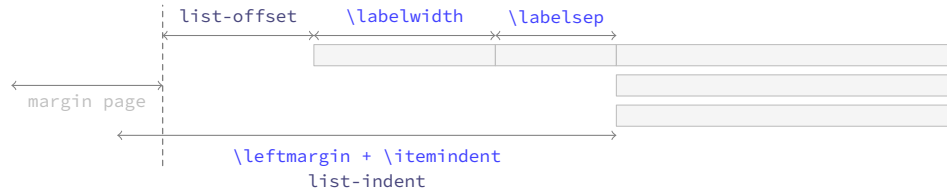


Figure 9: Representation of standard horizontal lengths in `list` environment.

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.

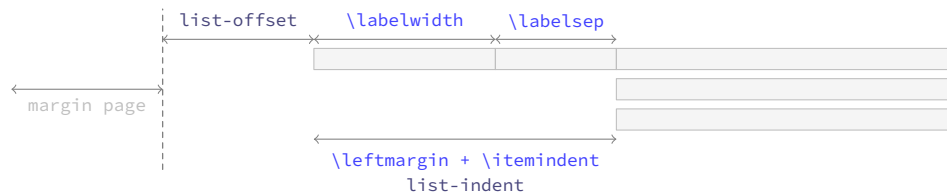


Figure 10: Representation of horizontal lengths concept in list in `enumext`.

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

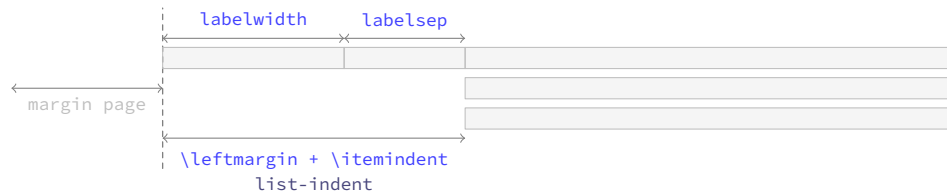


Figure 11: Default horizontal lengths in `enumext`.

```
__enumext_calc_hspace:NNNNNNN
__enumext_calc_hspace:ccccccc
```

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 (§13.38).

```
3534 \cs_new_protected:Npn __enumext_calc_hspace:NNNNNNN #1 #2 #3 #4 #5 #6 #7
3535 {
3536 \dim_compare:nNt { #1 } < { \c_zero_dim }
3537 {
3538 \msg_warning:nnnV { enumext } { width-non-positive } { labelwidth } { #1 }
3539 \dim_set:Nn #1 { \dim_abs:n { #1 } }
3540 }
3541 \dim_compare:nNt { #2 } < { \c_zero_dim }
3542 {
3543 \msg_warning:nnnV { enumext } { width-negative } { labelsep } { #2 }
3544 \dim_set:Nn #2 { \dim_abs:n { #2 } }
3545 }
3546 }
```

If no value has been passed to the `labelwidth` and `labelsep` keys we set the default values for `\l__enumext_leftmargin_tmp_X_dim`.

```
3546 \bool_if:nF #7 { \dim_set:Nn #4 { #1 + #2 } }
```

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

```

3547 \dim_compare:nNnTF { #4 } < { \c_zero_dim }
3548 {
3549 \dim_set:Nn #6 { #1 + #2 - #4 }
3550 \dim_set:Nn #5 { #1 + #2 + #3 - #6 }
3551 }
3552 {
3553 \dim_compare:nNnT { #4 } = { #1 + #2 }
3554 { \dim_set:Nn #6 { \c_zero_dim } }
3555 \dim_compare:nNnT { #4 } < { #1 + #2 }
3556 { \dim_set:Nn #6 { #1 + #2 - #4 } }
3557 \dim_compare:nNnT { #4 } > { #1 + #2 }
3558 {
3559 \dim_set:Nn #6 { -#1 - #2 + #4 }
3560 \dim_set:Nn #6 { #6*-1 }
3561 }
3562 \dim_set:Nn #5 { #1 + #2 + #3 - #6 }
3563 }
3564 }
3565 \cs_generate_variant:Nn __enumext_calc_hspace:NNNNNNN { cccccc }

```

(End of definition for `\__enumext_calc_hspace:NNNNNNN`.)

### 13.38.2 Setting second argument of the lists

We will “not set” `\leftmargini`, `\leftmarginii`, `\leftmarginiii` or `\leftmarginiv`, in this case, we will directly set the parameters for vertical and horizontal list spacing per level.

```

__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:
3566 \cs_set_protected:Npn __enumext_tmp:n #1
3567 {
3568 \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
3569 {
3570 __enumext_calc_hspace:ccccc
3571 { __enumext_labelwidth_#1_dim } { __enumext_labelsep_#1_dim }
3572 { __enumext_listoffset_#1_dim } { __enumext_leftmargin_tmp_#1_dim }
3573 { __enumext_leftmargin_#1_dim } { __enumext_itemindent_#1_dim }
3574 { __enumext_leftmargin_tmp_#1_bool }
3575 \clist_map_inline:nn
3576 { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
3577 { \dim_set_eq:cc {###1} { __enumext_###1_#1_dim } }
3578 \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
3579 { \skip_set_eq:cc {###1} { __enumext_###1_#1_skip } }
3580 \usecounter { enumX#1 }
3581 \setcounter { enumX#1 } { \int_eval:n { \int_use:c { __enumext_start_#1_int } - 1 } }
3582 \str_if_eq:nnTF { #1 } { v }
3583 {
3584 __enumext_keyans_redefine_item:
3585 __enumext_keyans_make_label:
3586 __enumext_keyans_ref:
3587 __enumext_keyans_fake_item_indent:
3588 \bool_if:cT { __enumext_show_length_#1_bool }
3589 {
3590 \msg_term:nnnn { enumext } { list-lengths-not-nested } { v } { keyans }
3591 }
3592 }
3593 {
3594 __enumext_redefine_item:
3595 __enumext_make_label:
3596 __enumext_standar_ref:
3597 __enumext_fake_item_indent:
3598 \bool_if:cT { __enumext_show_length_#1_bool }
3599 {
3600 \msg_term:nnne { enumext } { list-lengths } { #1 }
3601 { \int_use:N __enumext_level_int }
3602 }
3603 }
3604 }
3605 }
3606 \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 `0pt`. 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 `listparindent` and `parsep` to set the value of `\parskip` locally.

```

3607 \cs_set_protected:Npn __enumext_tmp:n #1
3608 {
3609 \cs_new_protected:cpn { __enumext_list_arg_two_#1: }
3610 {
3611 \bool_set_true:c { l__enumext_leftmargin_tmp_#1_bool }
3612 \dim_zero:c { l__enumext_leftmargin_tmp_#1_dim }
3613 __enumext_calc_hspace:cccccc
3614 { l__enumext_labelwidth_#1_dim } { l__enumext_labelsep_#1_dim }
3615 { l__enumext_listoffset_#1_dim } { l__enumext_leftmargin_tmp_#1_dim }
3616 { l__enumext_leftmargin_#1_dim } { l__enumext_itemindent_#1_dim }
3617 { l__enumext_leftmargin_tmp_#1_bool }
3618 \clist_map_inline:nn
3619 { labelsep, labelwidth, itemindent, leftmargin, rightmargin, listparindent }
3620 { \dim_set_eq:cc {####1} { l__enumext_####1_#1_dim } }
3621 \clist_map_inline:nn { topsep, parsep, partopsep, itemsep }
3622 { \skip_set_eq:cc {####1} { l__enumext_####1_#1_skip } }
3623 \skip_set_eq:Nc \parsep { l__enumext_itemsep_#1_skip }
3624 \skip_zero:N \partopsep
3625 \usecounter { enumX#1 }
3626 \setcounter { enumX#1 } { \int_eval:n { \int_use:c { l__enumext_start_#1_int } - 1 } }
3627 __enumext_starred_ref:
3628 \str_if_eq:nnTF {#1} { vii }
3629 {
3630 __enumext_fake_item_indent_vii:
3631 \bool_if:cT { l__enumext_show_length_vii_bool }
3632 { \msg_term:nnnn { enumext } { list-lengths-not-nested } { vii } { enumext* } }
3633 }
3634 {
3635 __enumext_fake_item_indent_viii:
3636 \bool_if:cT { l__enumext_show_length_#1_bool }
3637 { \msg_term:nnnn { enumext } { list-lengths-not-nested } { #1 } { keyans* } }
3638 }
3639 }
3640 }
3641 \clist_map_inline:nn { vii, viii } { __enumext_tmp:n {#1} }

```

(End of definition for `\__enumext_list_arg_two_vii:` and `\__enumext_list_arg_two_viii:`)

### 13.39 The environment `enumext`

```
__enumext_safe_exec:
```

The `\__enumext_safe_exec:` function first call the function `\__enumext_is_not_nested:` which sets `\g__enumext_standar_bool` to “true” if we are NOT nested within `enumext*`, then call the function `\__enumext_internal_mini_page:` to create the environment `__enumext_mini_page`, 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”.

```

3642 \cs_new_protected:Nn __enumext_safe_exec:
3643 {
3644 __enumext_is_not_nested:
3645 __enumext_internal_mini_page:
3646 \int_incr:N \l__enumext_level_int
3647 \int_compare:nNnT { \l__enumext_level_int } > { 4 }
3648 { \msg_fatal:nn { enumext } { list-too-deep } }
3649 \bool_set_true:N \l__enumext_standar_bool
3650 \bool_set_false:N \l__enumext_starred_bool
3651 __enumext_is_on_first_level:
3652 }

```

(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 `(keys)` 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 `(keys)` to the inner levels of the environment then we execute the function `\__enumext_store_active_keys:n` and reprocess the `(keys)` to pass them to the `sequence` if the key `save-key` is not active.

```

3653 \cs_new_protected:Npn __enumext_parse_keys:n #1
3654 {
3655 \tl_if_novalue:nF {#1}
3656 {
3657 \str_clear:N __enumext_series_str
3658 \int_compare:nNnTF { __enumext_level_int } = { 1 }
3659 {
3660 \keys_set:nn { enumext / level-1 } {#1}
3661 __enumext_parse_series:n {#1}
3662 __enumext_nested_base_line_fix:
3663 }
3664 {
3665 \exp_args:Ne \keys_set:nn
3666 { enumext / level-\int_use:N __enumext_level_int } {#1}
3667 }
3668 __enumext_store_active_keys:n {#1}
3669 }
3670 }

```

(End of definition for \\_\_enumext\_parse\_keys:n.)

\\_\_enumext\_start\_store\_level: The \\_\_enumext\_start\_store\_level: function activate the “*storing structure*” mechanism in the *sequence* for the command \anskey and the environment anskey\*.

```

3671 \cs_new_protected:Nn __enumext_start_store_level:
3672 {
3673 \bool_lazy_all:nT
3674 {
3675 { \bool_if_p:N __enumext_store_active_bool }
3676 { \bool_not_p:n { __enumext_keyans_env_bool } }
3677 { \bool_if_p:N \g__enumext_standar_bool }
3678 }
3679 {
3680 \int_compare:nNnT { __enumext_level_int } > { 1 }
3681 {
3682 \bool_set_true:c { l__enumext_store_upper_level_ __enumext_level: _bool }
3683 __enumext_store_level_open:
3684 }
3685 }

```

If enumext are nested in enumext\* add \\_\_enumext\_store\_level\_open: to preserve the “*storing structure*”.

```

3686 \bool_lazy_all:nT
3687 {
3688 { \bool_if_p:N __enumext_store_active_bool }
3689 { \bool_not_p:n { __enumext_keyans_env_bool } }
3690 { \int_compare_p:nNn { __enumext_level_h_int } = { 1 } }
3691 }
3692 {
3693 \int_compare:nNnT { __enumext_level_int } > { 0 }
3694 {
3695 \bool_set_true:c { l__enumext_store_upper_level_ __enumext_level: _bool }
3696 __enumext_store_level_open:
3697 }
3698 }
3699 }

```

(End of definition for \\_\_enumext\_start\_store\_level:.)

\\_\_enumext\_stop\_store\_level: The \\_\_enumext\_stop\_store\_level: function stop the “*storing structure*” mechanism in the *sequence* for the command \anskey and the environment anskey\*.

```

3700 \cs_new_protected:Nn __enumext_stop_store_level:
3701 {
3702 \bool_if:cT { l__enumext_store_upper_level_ __enumext_level: _bool }
3703 {
3704 __enumext_store_level_close:
3705 }
3706 }

```

(End of definition for \\_\_enumext\_stop\_store\_level:.)

`\__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=opt` and set the value of `\multicolsep` equal to zero and leave `\columnseprule` equal to zero for inner levels.

```

3707 \cs_new_protected:Nn __enumext_multicols_start:
3708 {
3709 \int_compare:nNt
3710 { \int_use:c { l__enumext_columns_ __enumext_level: _int } } > { 1 }
3711 {
3712 \dim_compare:nNt
3713 { \dim_use:c { l__enumext_columns_sep_ __enumext_level: _dim } } = { \c_zero_dim }
3714 {
3715 \dim_set:cn { l__enumext_columns_sep_ __enumext_level: _dim }
3716 {
3717 (\dim_use:c { l__enumext_labelwidth_ __enumext_level: _dim }
3718 + \dim_use:c { l__enumext_labelsep_ __enumext_level: _dim }
3719) / \int_use:c { l__enumext_columns_ __enumext_level: _int }
3720 - \dim_use:c { l__enumext_listoffset_ __enumext_level: _dim }
3721 }
3722 }
3723 \dim_set_eq:Nc \columnsep { l__enumext_columns_sep_ __enumext_level: _dim }
3724 \int_compare:nNt { \l__enumext_level_int } > { 1 }
3725 {
3726 \dim_zero:N \columnseprule
3727 }
3728 }
3729 }

```

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.

```

3728 \bool_if:cF { l__enumext_minipage_active_ __enumext_level: _bool }
3729 {
3730 \skip_zero:N \multicolsep
3731 __enumext_multi_addvspace:
3732 }
3733 \raggedcolumns
3734 \begin{multicols}{ \int_use:c { l__enumext_columns_ __enumext_level: _int } }
3735 }
3736 }

```

(End of definition for `\__enumext_multicols_start:`)

`\__enumext_multicols_stop:` The function `\__enumext_multicols_stop:` will stop the `multicols` environment and apply our “*vertical adjust*” spacing. For compatibility with *tagged* PDF, the closing of the `list` environment is executed here along with `\__enumext_stop_store_level:`.

```

3737 \cs_new_protected:Nn __enumext_multicols_stop:
3738 {
3739 \int_compare:nNtF
3740 { \int_use:c { l__enumext_columns_ __enumext_level: _int } } > { 1 }
3741 {
3742 __enumext_stop_list:
3743 __enumext_stop_store_level:
3744 \end{multicols}
3745 __enumext_unskip_unkern:
3746 __enumext_unskip_unkern:
3747 \par\addvspace{ \skip_use:c { l__enumext_multicols_below_ __enumext_level: _skip } }
3748 }
3749 {
3750 __enumext_stop_list:
3751 __enumext_stop_store_level:
3752 }
3753 }

```

(End of definition for `\__enumext_multicols_stop:`)

`\__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.

```

3754 \cs_new_protected:Nn __enumext_before_list:
3755 {
3756 __enumext_vspace_above:
3757 __enumext_before_args_exec:
3758 __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_page` 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_page` environment on the “*left side*”, always having a current `\linewidth` as *maximum width* between them.

```

3759 \dim_compare:nNt
3760 { \dim_use:c { \l__enumext_minipage_right_ __enumext_level: _dim } } > { \c_zero_dim }
3761 {
3762 \dim_set:cn { \l__enumext_minipage_left_ __enumext_level: _dim }
3763 {
3764 \linewidth
3765 - \dim_use:c { \l__enumext_minipage_right_ __enumext_level: _dim }
3766 - \dim_use:c { \l__enumext_minipage_hsep_ __enumext_level: _dim }
3767 }

```

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_minipage_add_space:` is called and the `__enumext_mini_page` 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.

```

3768 \bool_set_true:c { \l__enumext_minipage_active_ __enumext_level: _bool }
3769 \int_gincr:N \g__enumext_minipage_stat_int
3770 __enumext_minipage_add_space:
3771 \noindent
3772 __enumext_mini_page{ \dim_use:c { \l__enumext_minipage_left_ __enumext_level: _dim } }
3773 }
3774 __enumext_multicols_start:
3775 }

```

(End of definition for `\__enumext_before_list:`)

`\__enumext_second_part:` The function `\__enumext_second_part:` 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_page` environment has not been closed), then close `__enumext_mini_page` and add the *adjusted vertical space* `\l__enumext_minipage_after_skip`, otherwise we will close the `multicols` environment.

```

3776 \cs_new_protected:Nn __enumext_second_part:
3777 {
3778 \bool_if:cTF { \l__enumext_minipage_active_ __enumext_level: _bool }
3779 {
3780 \int_compare:nNt { \g__enumext_minipage_stat_int } = { 1 }
3781 {
3782 \msg_warning:nn { enumext } { missing-miniright }
3783 \miniright
3784 }
3785 \int_gzero:N \g__enumext_minipage_stat_int
3786 __enumext_unskip_unkern: % remove topsep + [partopsep]
3787 \end__enumext_mini_page
3788 }
3789 {
3790 __enumext_multicols_stop:
3791 }

```

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.

```

3792 __enumext_after_stop_list:
3793 __enumext_check_ans_key_hook:
3794 __enumext_vspace_below:
3795 \bool_set_false:N \l__enumext_standar_bool
3796 __enumext_resume_save_counter:
3797 }

```

(End of definition for `\__enumext_second_part:`)

`\__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.

```

3798 \cs_new_protected:Nn __enumext_set_item_width:

```

```

3799 {
3800 \dim_set:Nn \itemwidth { \linewidth }
3801 \dim_compare:nT
3802 {
3803 \dim_use:c { __enumext_listoffset_ __enumext_level: _dim } != \c_zero_dim
3804 }
3805 {
3806 \dim_sub:Nn \itemwidth
3807 {
3808 \dim_use:c { __enumext_listoffset_ __enumext_level: _dim }
3809 }
3810 }
3811 }

```

(End of definition for \\_\_enumext\_set\_item\_width:.)

**enumext** Now create the **enumext** environment based on **list** environment by levels.

```

3812 \NewDocumentEnvironment{enumext}{0}{ }
3813 {
3814 __enumext_safe_exec:
3815 __enumext_parse_keys:n {#1}
3816 __enumext_before_list:
3817 __enumext_start_store_level:
3818 __enumext_start_list:nn
3819 { \tl_use:c { __enumext_label_ __enumext_level: _tl } }
3820 {
3821 \use:c { __enumext_list_arg_two_ __enumext_level: : }
3822 __enumext_before_keys_exec:
3823 }
3824 __enumext_set_item_width:
3825 __enumext_after_args_exec:
3826 }
3827 {
3828 __enumext_second_part:
3829 }

```

(End of definition for enumext. This function is documented on page 5.)

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.

```

3830 __enumext_after_env:nn {enumext}
3831 {
3832 __enumext_execute_after_env:
3833 }

```

### 13.40 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”.

\\_\_enumext\_keyans\_safe\_exec:

The **keyans** environment will only be available if the **save-ans** key is active and can only be used at the “first level” within the **enumext** environment. We do not want the environment to be nested, so we will set a maximum at this point. If the conditions are not met, an error message will be returned.

```

3834 \cs_new_protected:Nn __enumext_keyans_safe_exec:
3835 {
3836 \bool_if:NF \l__enumext_store_active_bool
3837 {
3838 \msg_error:nnnn { enumext } { wrong-place } { keyans } { save-ans }
3839 }
3840 \int_incr:N \l__enumext_keyans_level_int
3841 \bool_set_true:N \l__enumext_keyans_env_bool
3842 __enumext_keyans_name_and_start:
3843 % Set false for interfering with enumext nested in keyans (yes, its possible and crayze)
3844 \bool_set_false:N \l__enumext_store_active_bool
3845 \int_compare:nNnT { \l__enumext_keyans_level_int } > { 1 }
3846 {
3847 \msg_error:nn { enumext } { keyans-nested }
3848 }
3849 \int_compare:nNnT { \l__enumext_level_int } > { 1 }
3850 {
3851 \msg_error:nn { enumext } { keyans-wrong-level }

```

```

3852 }
3853 }

```

(End of definition for `\__enumext_keyans_safe_exec:`)

```
__enumext_keyans_parse_keys:n
```

Parse [`<key = val>`] for `keyans` environment.

```

3854 \cs_new_protected:Npn __enumext_keyans_parse_keys:n #1
3855 {
3856 \keys_set:nn { enumext / keyans } {#1}
3857 }

```

(End of definition for `\__enumext_keyans_parse_keys:n`)

```
__enumext_before_list_v:
```

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

```
__enumext_keyans_multicols_start:
```

```
3858 \cs_new_protected:Nn __enumext_before_list_v:
```

```
__enumext_keyans_multicols_stop:
```

```
3859 {
```

```
__enumext_second_part_v:
```

```
3860 __enumext_vspace_above_v:
```

```
3861 __enumext_before_args_exec_v:
```

```
3862 \dim_compare:nNtT { \l__enumext_minipage_right_v_dim } > { \c_zero_dim }
```

```
3863 {
```

```
3864 \dim_set:Nn \l__enumext_minipage_left_v_dim
```

```
3865 {
```

```
3866 \linewidth - \l__enumext_minipage_right_v_dim - \l__enumext_minipage_hsep_v_dim
```

```
3867 }
```

```
3868 \bool_set_true:N \l__enumext_minipage_active_v_bool
```

```
3869 \int_gincr:N \g__enumext_minipage_stat_int
```

```
3870 __enumext_keyans_minipage_add_space:
```

```
3871 __enumext_mini_page{ \l__enumext_minipage_left_v_dim }
```

```
3872 }
```

```
3873 __enumext_keyans_multicols_start:
```

```
3874 }
```

```
3875 \cs_new_protected:Nn __enumext_keyans_multicols_start:
```

```
3876 {
```

```
3877 \int_compare:nNtT { \l__enumext_columns_v_int } > { 1 }
```

```
3878 {
```

```
3879 \dim_compare:nNtT { \l__enumext_columns_sep_v_dim } = { \c_zero_dim }
```

```
3880 {
```

```
3881 \dim_set:Nn \l__enumext_columns_sep_v_dim
```

```
3882 {
```

```
3883 (
```

```
3884 \l__enumext_labelwidth_v_dim + \l__enumext_labelsep_v_dim
```

```
3885) / \l__enumext_columns_v_int
```

```
3886 - \l__enumext_listoffset_v_dim
```

```
3887 }
```

```
3888 }
```

```
3889 \dim_set_eq:NN \columnsep \l__enumext_columns_sep_v_dim
```

```
3890 \dim_zero:N \columnseprule % no rule here
```

```
3891 \bool_if:NF \l__enumext_minipage_active_v_bool
```

```
3892 {
```

```
3893 \skip_zero:N \multicolsep
```

```
3894 __enumext_keyans_multi_addvspace:
```

```
3895 }
```

```
3896 \raggedcolumns
```

```
3897 \begin{multicols}{\l__enumext_columns_v_int }
```

```
3898 }
```

```
3899 }
```

```
3900 \cs_new_protected:Nn __enumext_keyans_multicols_stop:
```

```
3901 {
```

```
3902 \int_compare:nNtTF { \l__enumext_columns_v_int } > { 1 }
```

```
3903 {
```

```
3904 __enumext_stop_list:
```

```
3905 \end{multicols}
```

```
3906 __enumext_unskip_unkern:
```

```
3907 __enumext_unskip_unkern:
```

```
3908 \par\addvspace{ \l__enumext_multicols_below_v_skip }
```

```
3909 }
```

```
3910 {
```

```
3911 __enumext_stop_list:
```

```
3912 }
```

```
3913 }
```

```
3914 \cs_new_protected:Nn __enumext_second_part_v:
```

```

3915 {
3916 \bool_if:NTF \l__enumext_minipage_active_v_bool
3917 {
3918 \int_compare:nNt { \g__enumext_minipage_stat_int } = { 1 }
3919 {
3920 \msg_warning:nn { enumext } { missing-miniright }
3921 \miniright
3922 }
3923 \int_gzero:N \g__enumext_minipage_stat_int
3924 __enumext_unskip_unkern: % remove \topsep + [\partopsep]
3925 \end__enumext_mini_page
3926 \par\addvspace{ \l__enumext_minipage_after_skip }
3927 }
3928 {
3929 __enumext_keyans_multicols_stop:
3930 }
3931 \bool_set_false:N \l__enumext_keyans_env_bool
3932 __enumext_after_stop_list_v:
3933 __enumext_vspace_below_v:
3934 }

```

(End of definition for \\_\_enumext\_before\_list\_v: and others.)

\\_\_enumext\_keyans\_set\_item\_width:

The function \\_\_enumext\_keyans\_set\_item\_width: will set the value of \itemwidth taking into account the value established by the list-offset key.

```

3935 \cs_new_protected:Nn __enumext_keyans_set_item_width:
3936 {
3937 \dim_set:Nn \itemwidth { \linewidth }
3938 \dim_compare:nT
3939 {
3940 \l__enumext_listoffset_v_dim != \c_zero_dim
3941 }
3942 {
3943 \dim_sub:Nn \itemwidth { \l__enumext_listoffset_v_dim }
3944 }
3945 }

```

(End of definition for \\_\_enumext\_keyans\_set\_item\_width:.)

keyans

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

```

3946 \NewDocumentEnvironment{keyans}{ 0{} }
3947 {
3948 __enumext_keyans_safe_exec:
3949 __enumext_keyans_parse_keys:n {#1}
3950 __enumext_before_list_v:
3951 __enumext_start_list:nn
3952 { \tl_use:N \l__enumext_label_v_tl }
3953 {
3954 __enumext_list_arg_two_v:
3955 __enumext_before_keys_exec_v:
3956 }
3957 __enumext_keyans_set_item_width:
3958 __enumext_after_args_exec_v:
3959 }
3960 {
3961 __enumext_check_starred_cmd:n { item }
3962 __enumext_second_part_v:
3963 }

```

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

### 13.41 Tagging PDF support for non-standart list environments

The  $\TeX$  release 2022-06-01 brings automatic support for *tagged* PDF in several aspects, including the standard *list environments* and the `list` environment. Unfortunately non-standard *list environments* like `keyanspic` or the horizontal list environments `enumext*` and `keyans*` are not structured in a nice way, i.e. the expected result in the PDF file is the expected one, but the underlying structure is not correct. In simple terms, for *tagged* PDF a `list` environment is a `list` environment, no matter what it looks like in the PDF file.

To maintain a correct `list` structure when `\DocumentMetadata` is active, it is necessary to do some things manually using `tagpdf`[17] and `ltsockets`[19]. This implementation is an adaptation of my answer thanks to Ulrike Fischer's comments in [How can I modify my \item redefinition to be compatible with tagging-pdf.](#)



### 13.41.1 Socket for tagging support in enumext\* and keyans\*

```
start-list-tags
stop-start-tags
stop-list-tags
__enumext_start_list_tag:n
__enumext_stop_start_list_tag:
__enumext_stop_list_tag:n
```

We will first define the necessary sockets and their behavior for `enumext*` and `keyans*`.

```
3964 \socket_new:nn {tagsupport/__enumext/starred}{ 1 }
3965 \socket_new_plugin:nnn {tagsupport/__enumext/starred} {start-list-tags}
3966 {
3967 \tag_resume:n {#1}
3968 \tag_mc_end_push:
3969 \tag_struct_begin:n {tag=LI}
3970 \tag_struct_begin:n {tag=Lbl}
3971 \tag_mc_begin:n {tag=Lbl}
3972 }
3973 \socket_new_plugin:nnn {tagsupport/__enumext/starred} {stop-start-tags}
3974 {
3975 \tag_mc_end:
3976 \tag_struct_end:n {tag=Lbl}
3977 \tag_struct_begin:n {tag=LBody}
3978 \tag_struct_begin:n {tag=text-unit}
3979 \tag_struct_begin:n {tag=text}
3980 }
3981 \socket_new_plugin:nnn {tagsupport/__enumext/starred} {stop-list-tags}
3982 {
3983 \tag_struct_end:n {tag=text}
3984 \tag_struct_end:n {tag=text-unit}
3985 \tag_struct_end:n {tag=LBody}
3986 \tag_struct_end:n {tag=LI}
3987 \tag_mc_begin_pop:n {}
3988 \tag_suspend:n {#1}
3989 }
```

And now we'll wrap them so that they're only active when `\DocumentMetadata` is present.

```
3990 \cs_new_protected_nopar:Npn __enumext_start_list_tag:n #1
3991 {
3992 \IfDocumentMetadataTF
3993 {
3994 \socket_assign_plugin:nn {tagsupport/__enumext/starred} {start-list-tags}
3995 \socket_use:nn {tagsupport/__enumext/starred} {#1}
3996 } {}
3997 }
3998 \cs_new_protected_nopar:Npn __enumext_stop_start_list_tag:
3999 {
4000 \IfDocumentMetadataTF
4001 {
4002 \socket_assign_plugin:nn {tagsupport/__enumext/starred} {stop-start-tags}
4003 \socket_use:nn {tagsupport/__enumext/starred} {}
4004 } {}
4005 }
4006 \cs_new_protected_nopar:Npn __enumext_stop_list_tag:n #1
4007 {
4008 \IfDocumentMetadataTF
4009 {
4010 \socket_assign_plugin:nn {tagsupport/__enumext/starred} {stop-list-tags}
4011 \socket_use:nn {tagsupport/__enumext/starred} {#1}
4012 } {}
4013 }
```

*(End of definition for start-list-tags and others.)*

### 13.41.2 Socket for tagging support in keyanspic

```
start-list-tags
stop-start-tags
stop-list-tags
__enumext_anspic_start_list_tag:
__enumext_anspic_stop_start_list_tag:
__enumext_anspic_stop_list_tag:
```

We will first define the necessary sockets and their behavior for `keyanspic` environment.

```
4014 \socket_new:nn {tagsupport/__enumext/keyanspic}{ 0 }
4015 \socket_new_plugin:nnn {tagsupport/__enumext/keyanspic} {start-list-tags}
4016 {
4017 \tag_resume:n {keyanspic}
4018 \tag_mc_end_push:
4019 \tag_struct_begin:n {tag=LI}
4020 \tag_struct_begin:n {tag=Lbl}
4021 \tag_mc_begin:n {tag=Lbl}
4022 }
4023 \socket_new_plugin:nnn {tagsupport/__enumext/keyanspic} {stop-start-tags}
4024 {
4025 \tag_mc_end:
```

```
4026 \tag_struct_end:n {tag=Lbl}
4027 \tag_struct_begin:n {tag=LBody}
4028 \tag_struct_begin:n {tag=text-unit}
4029 \tag_struct_begin:n {tag=text}
4030 \tag_mc_begin:n {tag=text}
4031 }
4032 \socket_new_plug:nnn {tagsupport/__enumext/keyanspic} {stop-list-tags}
4033 {
4034 \tag_mc_end:
4035 \tag_struct_end:n {tag=text}
4036 \tag_struct_end:n {tag=text-unit}
4037 \tag_struct_end:n {tag=LBody}
4038 \tag_struct_end:n {tag=LI}
4039 \tag_mc_begin_pop:n {}
4040 \tag_suspend:n {keyanspic}
4041 }
```

And now we'll wrap them so that they're only active when `\DocumentMetadata` is present.

```
4042 \cs_new_protected_nopar:Nn __enumext_anspic_start_list_tag:
4043 {
4044 \IfDocumentMetadataTF
4045 {
4046 \socket_assign_plug:nn {tagsupport/__enumext/keyanspic} {start-list-tags}
4047 \socket_use:n {tagsupport/__enumext/keyanspic}
4048 } {}
4049 }
4050 \cs_new_protected_nopar:Nn __enumext_anspic_stop_start_list_tag:
4051 {
4052 \IfDocumentMetadataTF
4053 {
4054 \socket_assign_plug:nn {tagsupport/__enumext/keyanspic} {stop-start-tags}
4055 \socket_use:n {tagsupport/__enumext/keyanspic}
4056 } {}
4057 }
4058 \cs_new_protected_nopar:Nn __enumext_anspic_stop_list_tag:
4059 {
4060 \IfDocumentMetadataTF
4061 {
4062 \socket_assign_plug:nn {tagsupport/__enumext/keyanspic} {stop-list-tags}
4063 \socket_use:n {tagsupport/__enumext/keyanspic}
4064 } {}
4065 }
```

(End of definition for `start-list-tags` and others.)

13.42 The environment `keyanspic` and `\anspic`

The `keyanspic` environment is a `list` based environment that uses the same configuration for “*spacing*” and `<label>` as the `keyans` environment, but it does not use `\item`. The `<contents>` are passed to the environment by means of the `\anspic` command as replacement for `\item` command and placed inside `minipage` environments, with the `<label>` centered “*above*” or “*below*”, adjusting *widths* and *position* according to the options passed to the environment.

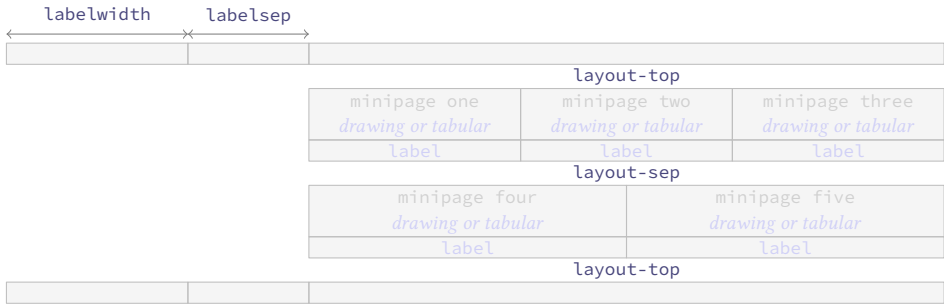


Figure 12: Representation of the `keyanspic` spacing in `enumext`.

In order for the `keyanspic` environment and the `\anspic` command to work correctly, we need to set and export some variables in the first part of the environment definition and pass them to `\anspic` which is executed in the second part of the environment. This implementation is adapted from the answer given by Enrico Gregorio (@egreg) in [How to process the body of an environment and divide it by a \macro?](#).

13.42.1 The environment `keyanspic`

`label-pos` First we define the key that allows us to process the position of the `<label>` centered “*above*” or “*below*” which will be `label-pos`, the vertical separation of these from `drawing or tabular` will be handled with the key

**label-sep.** The “*layout style*” will be handled with the key `layout-sty` will take two values separated by comma  $\{ \langle n^\circ \text{ upper}, n^\circ \text{ lower} \rangle \}$  and will determine the number of `minipage` environments in which all arguments of `\anspic` will be printed at the “upper” and “lower” within the environments separated by the value of the key `layout-sep`. The vertical space “top” and “bottom” of the environment will be handled with the key `layout-top`.

```

4066 \keys_define:nn { enumext / keyanspic }
4067 {
4068 label-pos .choice:,
4069 label-pos / above .code:n =
4070 \bool_set_true:N \l__enumext_anspic_label_above_bool
4071 \str_set:Nn \l__enumext_anspic_mini_pos_str { t },
4072 label-pos / below .code:n =
4073 \bool_set_false:N \l__enumext_anspic_label_above_bool
4074 \str_set:Nn \l__enumext_anspic_mini_pos_str { b },
4075 label-pos / unknown .code:n =
4076 \msg_error:nneee { enumext } { unknown-choice }
4077 { label-pos } { above,~ below } { \exp_not:n {#1} },
4078 label-pos .initial:n = below,
4079 label-pos .value_required:n = true,
4080 label-sep .skip_set:N = \l__enumext_anspic_label_sep_skip,
4081 label-sep .value_required:n = true,
4082 layout-sty .tl_set:N = \l__enumext_anspic_layout_style_tl,
4083 layout-sty .value_required:n = true,
4084 layout-sep .code:n = \keys_set:nn { enumext / keyans }
4085 { parsep = #1 },
4086 layout-sep .value_required:n = true,
4087 layout-top .code:n = \keys_set:nn { enumext / keyans }
4088 { topsep = #1 },
4089 layout-top .value_required:n = true,
4090 unknown .code:n = { \l__enumext_keyans_unknown_keys:n {#1} }
4091 }

```

(End of definition for `label-pos` and others.)

```

__enumext_keyans_pic_safe_exec:
__enumext_keyans_pic_parse_keys:n
__enumext_keyans_pic_skip_abs:N
__enumext_keyans_pic_arg_two:

```

The function `\__enumext_keyans_pic_safe_exec:` check the nested level position inside the `enumext` environment.

```

4092 \cs_new_protected:Nn __enumext_keyans_pic_safe_exec:
4093 {
4094 \int_incr:N \l__enumext_keyans_pic_level_int
4095 \int_compare:nNnT { \l__enumext_keyans_pic_level_int } > { 1 }
4096 {
4097 \msg_error:nn { enumext } { keyanspic-nested }
4098 }
4099 __enumext_keyans_name_and_start:
4100 }

```

Parse  $[\langle \textit{key} = \textit{val} \rangle]$  for `keyanspic` environment.

```

4101 \cs_new_protected:Npn __enumext_keyans_pic_parse_keys:n #1
4102 {
4103 \tl_if_novalue:nF {#1}
4104 {
4105 \keys_set:nn { enumext / keyanspic } {#1}
4106 }
4107 }

```

The function `\__enumext_keyans_pic_skip_abs:N` will return a positive value `\parsep` from `keyans` environment.

```

4108 \cs_new_protected:Npn __enumext_keyans_pic_skip_abs:N #1
4109 {
4110 \dim_compare:nNnT { #1 } < { \c_zero_dim }
4111 {
4112 \skip_set:Nn #1 { -#1 }
4113 }
4114 }

```

The `\__enumext_keyans_pic_arg_two:` function will be used in the *second argument* of the `list` environment that defines the `keyanspic` environment, with this we will take the configuration of the “*spaces*” and the keys `label`, `wrap-label`, `parsep` and `topsep` from the `keyans` environment. The first thing we need to do is set the boolean variable `\l__enumext_leftmargin_tmp_v_bool` handled by the `list-indent` key to “false”, then copy the definition of the second list argument from the `keyans` environment definition and make sure that `\parsep` does not have a negative value.

```

4115 \cs_new_protected:Npn __enumext_keyans_pic_arg_two:
4116 {
4117 \bool_set_false:N \l__enumext_leftmargin_tmp_v_bool
4118 __enumext_list_arg_two_v:
4119 __enumext_keyans_pic_skip_abs:N \parsep

```

Now we increment the counter `enumXv` of the `keyans` environment and save the *total height* of the *(label)* in `\l__enumext_anspic_label_htdp_dim` used by `\anspic` and we will adjust the values of `\parsep` only if the key `label-pos` is set to *below*.

```

4120 \bool_if:NF \l__enumext_anspic_label_above_bool
4121 {
4122 \stepcounter { enumXv }
4123 \hbox_set:Nn \l__enumext_anspic_label_box { \l__enumext_label_v_tl }
4124 \dim_set:Nn \l__enumext_anspic_label_htdp_dim
4125 {
4126 \box_ht_plus_dp:N \l__enumext_anspic_label_box
4127 }
4128 \skip_add:Nn \parsep
4129 {
4130 \l__enumext_anspic_label_htdp_dim
4131 + \box_dp:N \strutbox
4132 + \l__enumext_anspic_label_sep_skip
4133 }
4134 }

```

Finally we *adjust* the value of `\leftmargin` and `\topsep` then set `\listparindent`, `\partopsep` and `\itemsep` to zero so that the *horizontal* and *vertical* space is not affected.

```

4135 \dim_add:Nn \leftmargin { -\labelwidth - \labelsep }
4136 \skip_add:Nn \topsep { 0.5\box_dp:N \strutbox }
4137 \dim_zero:N \listparindent
4138 \skip_zero:N \partopsep
4139 \skip_zero:N \itemsep
4140 }

```

(End of definition for `\__enumext_keyans_pic_safe_exec:` and others.)

**keyanspic** Now we define the environment `keyanspic`. For compatibility with *tagged* PDF we must use the `\begin{list}` form and a lot of conditional code using `\IfDocumentMetadataTF`. We will first stop the code for automatic *tagged* PDF for `list` environments, redefine `\item` so that it cannot be used, and stop the code for automatic *tagged* PDF for the `keyanspic` environment.

```

4141 \NewDocumentEnvironment{keyanspic}{ o }
4142 {
4143 __enumext_keyans_pic_safe_exec:
4144 __enumext_keyans_pic_parse_keys:n {#1}
4145 \begin{list} { } { __enumext_keyans_pic_arg_two: }
4146 \IfDocumentMetadataTF
4147 {
4148 \tag_suspend:n {list}
4149 }{}
4150 \item[] \scan_stop:
4151 \RenewDocumentCommand \item { }
4152 {
4153 \msg_error:nn { enumext } { keyanspic-item-cmd }
4154 }
4155 \IfDocumentMetadataTF
4156 {
4157 \tag_resume:n {keyanspic}
4158 \tag_tool:n {para/tagging=false}
4159 \tag_suspend:n {keyanspic}
4160 } { }
4161 }

```

In the second part of the environment definition we will manually place our code for *tagged* PDF and execute the command `\anspic` using the `\__enumext_anspic_exec:` function.

```

4162 {
4163 \IfDocumentMetadataTF
4164 {
4165 \tag_resume:n {keyanspic}
4166 \tag_mc_end_push:
4167 \tag_struct_begin:n {tag=L,attribute=enumerate}
4168 } { }
4169 __enumext_anspic_exec:

```

```

4170 \IfDocumentMetadataTF
4171 {
4172 \tag_suspend:n {keyanspic}
4173 } { }
4174 \end{list}
4175 \IfDocumentMetadataTF
4176 {
4177 \tag_struct_end:n {tag=L}
4178 \tag_mc_begin_pop:n {}
4179 \tag_struct_end:n {tag=L}
4180 \tag_mc_begin_pop:n {}
4181 } { }

```

Finally we check if `\anspic*` has been used, set the counter `enumXvi` to zero and apply our “adjusted” vertical space bottom.

```

4182 __enumext_check_starred_cmd:n { anspic }
4183 \setcounter { enumXvi } { 0 }
4184 \bool_if:NTF \l__enumext_anspic_label_above_bool
4185 {
4186 \par\addvspace{ 0.5\box_dp:N \strutbox }
4187 }
4188 {
4189 \par
4190 \addvspace
4191 {
4192 \dim_eval:n
4193 {
4194 \l__enumext_anspic_label_htdp_dim + \box_ht_plus_dp:N \strutbox
4195 + \l__enumext_anspic_label_sep_skip + \l__enumext_topsep_v_skip
4196 }
4197 }
4198 }
4199 }

```

(End of definition for `keyanspic`. This function is documented on page 15.)

### 13.42.2 The command `\anspic`

The `\anspic` command take three arguments, the *starred versions* `\anspic*[\langle content \rangle]` store the current `\label` next to the *optional argument* `[\langle content \rangle]` in the *sequence* and *prop list* defined by `save-ans` key. The third *mandatory argument* `{\langle drawing or tabular \rangle}` is NOT stored in the *sequence* or *prop list*.

- One of the complications here to make the `keyanspic` environment compatible with *tagged* PDF is the position of `\label`, the `\anspic` command processes the arguments in order, where #1 and #2 correspond to `\label` and #3 to the mandatory argument and puts all this inside a `minipage` environment. If #1 and #2, that is `\label`, is above #3 there are no problems with *tagged* PDF, but if #3 comes first the list created with *tagged* PDF will not be correct.

`\anspic`

We check that the command is active in the `keyanspic` environment only if the `save-ans` key is present, otherwise we return an error. The three arguments are handled by the function `\__enumext_anspic_args:nnn` and stored in the sequence `\l__enumext_anspic_args_seq` which is processed by the `keyanspic` environment.

```

4200 \NewDocumentCommand \anspic { s o +m }
4201 {
4202 \bool_if:NF \l__enumext_store_active_bool
4203 {
4204 \msg_error:nnnn { enumext } { wrong-place } { keyanspic } { save-ans }
4205 }
4206 \int_compare:nNt { \l__enumext_level_int } > { 1 }
4207 {
4208 \msg_error:nn { enumext } { keyanspic-wrong-level }
4209 }
4210 \int_compare:nNt { \l__enumext_keyans_level_int } = { 1 }
4211 {
4212 \msg_error:nnnn { enumext } { command-wrong-place } { anspic } { keyans }
4213 }
4214 \seq_put_right:Nn \l__enumext_anspic_args_seq
4215 {
4216 __enumext_anspic_args:nnn { #1 } { #2 } { #3 }
4217 }
4218 }

```

The `\__enumext_anspic_body_dim:n` function will set the value of `\l__enumext_anspic_body_htdp_dim` equal to the “height plus depth” of the *mandatory argument* if the key `label-pos` is set “below”.

```

4219 \cs_new_protected:Npn __enumext_anspic_body_dim:n #1
4220 {
4221 \bool_if:NF \l__enumext_anspic_label_above_bool
4222 {
4223 \IfDocumentMetadataTF
4224 {
4225 \tag_suspend:n {keyanspic}
4226 } { }
4227 \vbox_set:Nn \l__enumext_anspic_body_box { #1 }
4228 \dim_set:Nn \l__enumext_anspic_body_htdp_dim
4229 {
4230 \box_ht_plus_dp:N \l__enumext_anspic_body_box
4231 }
4232 \IfDocumentMetadataTF
4233 {
4234 \tag_resume:n {keyanspic}
4235 } { }
4236 }
4237 }

```

The `\__enumext_anspic_label:nn` function will process inside `\makebox` the *starred argument* ‘\*’ and *optional argument* passed to the command. Here we will store the *label* and *optional argument* in *prop list* and *sequence* and execute the `show-ans`, `show-pos`, `font`, `wrap-label` and `wrap-opt` keys.

```

4238 \cs_new_protected:Npn __enumext_anspic_label:nn #1 #2
4239 {
4240 \makebox[\l__enumext_anspic_mini_width_dim][c]
4241 {
4242 \bool_if:nT { #1 }
4243 {
4244 __enumext_keyans_addto_prop:n { #2 }
4245 __enumext_keyans_store_ref:
4246 __enumext_keyans_addto_seq:n { #2 }
4247 \int_gincr:N \g__enumext_check_starred_cmd_int
4248 \bool_lazy_or:nnT
4249 { \bool_if_p:N \l__enumext_show_answer_bool }
4250 { \bool_if_p:N \l__enumext_show_position_bool }
4251 {
4252 \tl_set_eq:NN \l__enumext_label_v_tl \l__enumext_label_vi_tl
4253 __enumext_keyans_show_left:n { #2 }
4254 \tl_set_eq:NN \l__enumext_label_vi_tl \l__enumext_label_v_tl
4255 }
4256 }
4257 \tl_use:N \l__enumext_label_font_style_v_tl
4258 __enumext_wrapper_label_v:n { \l__enumext_label_vi_tl }
4259 __enumext_keyans_show_item_opt:
4260 }
4261 }

```

The function `\__enumext_anspic_label_pos:nnn` will be in charge of handling the “*counter*” and the position of the *label*, set by `label-pos` key which will have the same configuration as the `keyans` environment.

```

4262 \cs_new_protected:Npn __enumext_anspic_label_pos:nnn #1 #2 #3
4263 {
4264 \stepcounter { enumXvi }
4265 __enumext_anspic_body_dim:n { #3 }
4266 \bool_if:NTF \l__enumext_anspic_label_above_bool
4267 {
4268 __enumext_anspic_label:nn { #1 } { #2 }
4269 }
4270 {
4271 \raisebox
4272 {
4273 -\dim_eval:n
4274 {
4275 \l__enumext_anspic_label_htdp_dim
4276 + \l__enumext_anspic_body_htdp_dim
4277 + \box_dp:N \strutbox
4278 + \l__enumext_anspic_label_sep_skip
4279 }
4280 }
4281 [opt] [opt]
4282 {
4283 __enumext_anspic_label:nn { #1 } { #2 }

```

```

4284 }
4285 }
4286 }
4287 %

```

The `\__enumext_anspic_args:nnn` function will be responsible for placing the code compatible with *tagged* PDF and the arguments within the `\l__enumext_anspic_args_seq` sequence which will be processed by the `\__enumext_anspic_print:n` function in the second part of the definition of the `keyanspic` environment.

```

4288 \cs_new_protected:Nn __enumext_anspic_args:nnn
4289 {
4290 __enumext_anspic_start_list_tag:
4291 __enumext_anspic_label_pos:nnn { #1 } { #2 } { #3 }
4292 __enumext_anspic_stop_start_list_tag:
4293 \bool_if:NTF \l__enumext_anspic_label_above_bool
4294 {
4295 \[\l__enumext_anspic_label_sep_skip] #3
4296 }
4297 {
4298 \[#3
4299 }
4300 __enumext_anspic_stop_list_tag:
4301 }

```

The value  $\{ \langle n^{\circ} upper, n^{\circ} lower \rangle \}$  passed to the `layout-sty` key is split by comma and is handled directly by the function `\__enumext_anspic_print:n` and passed to the function `\__enumext_anspic_row:n`.

```

4302 \cs_new_protected:Nn __enumext_anspic_print:n
4303 {
4304 \clist_map_function:nN { #1 } __enumext_anspic_row:n
4305 }
4306 \cs_generate_variant:Nn __enumext_anspic_print:n { e, V }

```

The function `\__enumext_anspic_row:n` will set the *widths* for the `minipage` environments and place *all arguments* passed to `\anspic` saved in the `\l__enumext_anspic_args_seq` sequence inside them.

```

4307 \cs_new_protected:Nn __enumext_anspic_row:n
4308 {
4309 \dim_set:Nn \l__enumext_anspic_mini_width_dim { \linewidth / #1 }
4310 \int_set:Nn \l__enumext_anspic_above_int { \l__enumext_anspic_below_int }
4311 \int_set:Nn \l__enumext_anspic_below_int { \l__enumext_anspic_above_int + #1 }
4312 \int_step_inline:nnn
4313 { \l__enumext_anspic_above_int + 1 }
4314 { \l__enumext_anspic_below_int }
4315 {
4316 \IfDocumentMetadataTF
4317 {
4318 \tag_suspend:n {minipage}
4319 } { }
4320 \begin{minipage}[\l__enumext_anspic_mini_pos_str]{ \l__enumext_anspic_mini_width_dim }
4321 \centering
4322 \seq_item:Nn \l__enumext_anspic_args_seq { ##1 }
4323 \end{minipage}
4324 \IfDocumentMetadataTF
4325 {
4326 \tag_resume:n {minipage}
4327 } { }
4328 }
4329 \par
4330 }

```

The `\__enumext_anspic_exec:` function will execute all the code in the `\anspic` command in the second argument of the `keyanspic` environment definition. If the key `layout-sty` is not set, everything will be printed on a *single line*.

```

4331 \cs_new_protected:Nn __enumext_anspic_exec:
4332 {
4333 \tl_if_empty:NTF \l__enumext_anspic_layout_style_tl
4334 {
4335 __enumext_anspic_print:e { \seq_count:N \l__enumext_anspic_args_seq }
4336 }
4337 {
4338 __enumext_anspic_print:V \l__enumext_anspic_layout_style_tl
4339 }
4340 }

```

(End of definition for `\anspic` and others. This function is documented on page 16.)



### 13.43 The horizontal environments

Generating *horizontal list environments* is NOT as simple as standard  $\TeX$  list environments. The fundamental part of the code is adapted from the `shortlist` package to a more modern version using `expl3`. It is not possible to redefine `\item` and `\makeLabel` using `\RenewDocumentCommand` as in the vertical *non starred* versions.

To achieve the *horizontal list environments* we will capture the `\item` command and the  $\langle content \rangle$  of this in *horizontal box* using `\makebox` for the `label` and a `minipage` environment for the  $\langle content \rangle$  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 *first optional argument* ( $\langle number \rangle$ ).

A side effect is the limitation of using `\item` in this way *without* using `\RenewDocumentCommand`, which loses the original definition and affects the *standard list environments* provided by  $\TeX$  and any environment defined using base `list` environment, including: `itemize`, `enumerate`, `description`, `quote`, `quotation`, `verse`, `center`, `flushleft`, `flushright`, `verbatim`, `tabbing`, `trivlist`, `list` and all environments created with `\newtheorem`.

One way to get around this is to use something like:

```
\AddToHook{env/enumerate/before}{recover original \item definition}
```

inside `minipage`, but in my partial tests this does not have the desired effect and the vertical and horizontal spacing is distorted. For now this will remain as a limitation and I will see if it is feasible to implement it in the future.

For compatibility with the *tagged* PDF we close the environments according to the presence or not of the `mini-env` key.

#### 13.43.1 Functions for item box width

We set the default value for the *width of the box* containing the  $\langle content \rangle$  of the items for `enumext*` environment.

```
__enumext_starred_columns_set_vii:
__enumext_starred_columns_set_viii:
```

```
4341 \cs_new_protected:Nn __enumext_starred_columns_set_vii:
4342 {
4343 \dim_compare:nNt { __enumext_columns_sep_vii_dim } = { \c_zero_dim }
4344 {
4345 \dim_set:Nn __enumext_columns_sep_vii_dim
4346 {
4347 (__enumext_labelwidth_vii_dim + __enumext_labelsep_vii_dim)
4348 / __enumext_columns_vii_int
4349 }
4350 }
4351 \int_set:Nn __enumext_tmpa_vii_int { __enumext_columns_vii_int - 1 }
4352 \dim_set:Nn __enumext_item_width_vii_dim
4353 {
4354 (\linewidth - __enumext_columns_sep_vii_dim * __enumext_tmpa_vii_int)
4355 / __enumext_columns_vii_int
4356 - __enumext_labelwidth_vii_dim
4357 - __enumext_labelsep_vii_dim
4358 }
4359 }
```

When the key `rightmargin` is active we must adjust the values.

```
4359 \dim_compare:nNt { __enumext_rightmargin_vii_dim } > { \c_zero_dim }
4360 {
4361 \dim_sub:Nn __enumext_item_width_vii_dim
4362 {
4363 (__enumext_rightmargin_vii_dim * __enumext_tmpa_vii_int)
4364 / __enumext_columns_vii_int
4365 }
4366 \dim_add:Nn __enumext_columns_sep_vii_dim
4367 {
4368 __enumext_rightmargin_vii_dim
4369 }
4370 }
4371 }
```

Same implementation for the `keyans*` environment.

```
4372 \cs_new_protected:Nn __enumext_starred_columns_set_viii:
4373 {
4374 \dim_compare:nNt { __enumext_columns_sep_viii_dim } = { \c_zero_dim }
4375 {
4376 \dim_set:Nn __enumext_columns_sep_viii_dim
4377 {
4378 (__enumext_labelwidth_viii_dim + __enumext_labelsep_viii_dim)
4379 / __enumext_columns_viii_int
4380 }
4381 }
4382 }
```

```

4382 \int_set:Nn \l__enumext_tmpa_viii_int { \l__enumext_columns_viii_int - 1 }
4383 \dim_set:Nn \l__enumext_item_width_viii_dim
4384 {
4385 (\linewidth - \l__enumext_columns_sep_viii_dim * \l__enumext_tmpa_viii_int)
4386 / \l__enumext_columns_viii_int
4387 - \l__enumext_labelwidth_viii_dim
4388 - \l__enumext_labelsep_viii_dim
4389 }
4390 \dim_compare:nNnT { \l__enumext_rightmargin_viii_dim } > { \c_zero_dim }
4391 {
4392 \dim_sub:Nn \l__enumext_item_width_viii_dim
4393 {
4394 (\l__enumext_rightmargin_viii_dim * \l__enumext_tmpa_vii_int)
4395 / \l__enumext_columns_viii_int
4396 }
4397 \dim_add:Nn \l__enumext_columns_sep_viii_dim
4398 {
4399 \l__enumext_rightmargin_viii_dim
4400 }
4401 }
4402 }

```

(End of definition for `\__enumext_starred_columns_set_vii:` and `\__enumext_starred_columns_set_viii:`)

### 13.43.2 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(<columns>)` will be stored together with the value of `\itemwidth` for the `enumext*` environment.

```

4403 \cs_new_protected:Npn __enumext_starred_joined_item_vii:n #1
4404 {
4405 \int_set:Nn \l__enumext_joined_item_vii_int {#1}
4406 \int_compare:nNnT { \l__enumext_joined_item_vii_int } > { \l__enumext_columns_vii_int }
4407 {
4408 \msg_warning:nnee { enumext } { item-joined }
4409 { \int_use:N \l__enumext_joined_item_vii_int }
4410 { \int_use:N \l__enumext_columns_vii_int }
4411 \int_set:Nn \l__enumext_joined_item_vii_int
4412 {
4413 \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1
4414 }
4415 }
4416 \int_compare:nNnT
4417 { \l__enumext_joined_item_vii_int }
4418 >
4419 { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1 }
4420 {
4421 \msg_warning:nnee { enumext } { item-joined-columns }
4422 { \int_use:N \l__enumext_joined_item_vii_int }
4423 {
4424 \int_eval:n
4425 { \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1 }
4426 }
4427 \int_set:Nn \l__enumext_joined_item_vii_int
4428 {
4429 \l__enumext_columns_vii_int - \l__enumext_item_column_pos_vii_int + 1
4430 }
4431 }
4432 \int_compare:nNnTF { \l__enumext_joined_item_vii_int } > { 1 }
4433 {
4434 \int_set_eq:NN \l__enumext_joined_item_aux_vii_int \l__enumext_joined_item_vii_int
4435 \int_decr:N \l__enumext_joined_item_aux_vii_int
4436 \int_add:Nn \l__enumext_item_column_pos_vii_int { \l__enumext_joined_item_aux_vii_int }
4437 \int_gadd:Nn \g__enumext_item_count_all_vii_int { \l__enumext_joined_item_aux_vii_int }
4438 \dim_set:Nn \l__enumext_joined_width_vii_dim
4439 {
4440 \l__enumext_item_width_vii_dim * \l__enumext_joined_item_vii_int
4441 + (\l__enumext_labelwidth_vii_dim + \l__enumext_labelsep_vii_dim
4442 + \l__enumext_columns_sep_vii_dim
4443) * \l__enumext_joined_item_aux_vii_int
4444 }
4445 \dim_set_eq:NN \itemwidth \l__enumext_joined_width_vii_dim

```

```

4446 }
4447 {
4448 \dim_set_eq:NN \l__enumext_joined_width_vii_dim \l__enumext_item_width_vii_dim
4449 \dim_set_eq:NN \itemwidth \l__enumext_item_width_vii_dim
4450 }
4451 }

```

Same implementation for the **keyans\*** environment.

```

4452 \cs_new_protected:Npn __enumext_starred_joined_item_viii:n #1
4453 {
4454 \int_set:Nn \l__enumext_joined_item_viii_int {#1}
4455 \int_compare:nNnT { \l__enumext_joined_item_viii_int } > { \l__enumext_columns_viii_int }
4456 {
4457 \msg_warning:nnee { enumext } { item-joined }
4458 { \int_use:N \l__enumext_joined_item_viii_int }
4459 { \int_use:N \l__enumext_columns_viii_int }
4460 \int_set:Nn \l__enumext_joined_item_viii_int
4461 {
4462 \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1
4463 }
4464 }
4465 \int_compare:nNnT
4466 { \l__enumext_joined_item_viii_int }
4467 >
4468 { \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1 }
4469 {
4470 \msg_warning:nnee { enumext } { item-joined-columns }
4471 { \int_use:N \l__enumext_joined_item_viii_int }
4472 {
4473 \int_eval:n
4474 { \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1 }
4475 }
4476 \int_set:Nn \l__enumext_joined_item_viii_int
4477 {
4478 \l__enumext_columns_viii_int - \l__enumext_item_column_pos_viii_int + 1
4479 }
4480 }
4481 \int_compare:nNnTF { \l__enumext_joined_item_viii_int } > { 1 }
4482 {
4483 \int_set_eq:NN \l__enumext_joined_item_aux_viii_int \l__enumext_joined_item_viii_int
4484 \int_decr:N \l__enumext_joined_item_aux_viii_int
4485 \int_add:Nn \l__enumext_item_column_pos_viii_int { \l__enumext_joined_item_aux_viii_int }
4486 \int_gadd:Nn \g__enumext_item_count_all_viii_int { \l__enumext_joined_item_aux_viii_int }
4487 \dim_set:Nn \l__enumext_joined_width_viii_dim
4488 {
4489 \l__enumext_item_width_viii_dim * \l__enumext_joined_item_viii_int
4490 + (\l__enumext_labelwidth_viii_dim + \l__enumext_labelsep_viii_dim
4491 + \l__enumext_columns_sep_viii_dim
4492) * \l__enumext_joined_item_aux_viii_int
4493 }
4494 \dim_set_eq:NN \itemwidth \l__enumext_joined_width_viii_dim
4495 }
4496 {
4497 \dim_set_eq:NN \l__enumext_joined_width_viii_dim \l__enumext_item_width_viii_dim
4498 \dim_set_eq:NN \itemwidth \l__enumext_item_width_viii_dim
4499 }
4500 }

```

(End of definition for `\__enumext_starred_joined_item_vii:n` and `\__enumext_starred_joined_item_viii:n`)

### 13.43.3 Functions for mini-env, mini-right and mini-right\* keys

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_page` 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`.

```

4501 \cs_new_protected:Nn __enumext_start_mini_vii:
4502 {
4503 \dim_compare:nNnT { \l__enumext_minipage_right_vii_dim } > { \c_zero_dim }
4504 {
4505 \dim_set:Nn \l__enumext_minipage_left_vii_dim
4506 {

```

```

4507 \linewidth
4508 - \l__enumext_minipage_right_vii_dim
4509 - \l__enumext_minipage_hsep_vii_dim
4510 }
4511 \bool_set_true:N \l__enumext_minipage_active_vii_bool
4512 \dim_gset_eq:NN
4513 \g__enumext_minipage_right_vii_dim
4514 \l__enumext_minipage_right_vii_dim
4515 __enumext_mini_addvspace_vii:
4516 \nointerlineskip\noindent
4517 __enumext_mini_page{ \l__enumext_minipage_left_vii_dim }
4518 }
4519 }

```

The function `\__enumext_stop_mini_vii:` closes the `\__enumext_mini_page` environment on the “left side”, applies `\hfill` and set the variable `\g__enumext_minipage_active_vii_bool` to “true” which will be used in the function `\__enumext_after_env:nn` to execute the `minipage` on the “right side”. At this point we will execute the `\__enumext_stop_list:` and `\__enumext_stop_store_level_vii:` functions stopping the `list` environment and the level saving mechanism for storage in *sequence* of the `\anskey` command and `anskey*` environment. This function is passed to the `\__enumext_after_list_vii:` function in the second part of the `enumext*` environment definition (§13.44).

```

4520 \cs_new_protected:Nn __enumext_stop_mini_vii:
4521 {
4522 \bool_if:NTF \l__enumext_minipage_active_vii_bool
4523 {
4524 __enumext_stop_list:
4525 __enumext_stop_store_level_vii:
4526 \IfDocumentMetadataTF { \tag_resume:n {enumext*} } { }
4527 \end__enumext_mini_page
4528 \hfill
4529 \bool_gset_true:N \g__enumext_minipage_active_vii_bool
4530 }
4531 {
4532 __enumext_stop_list:
4533 __enumext_stop_store_level_vii:
4534 }
4535 }

```

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

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

```

4536 __enumext_after_env:nn {enumext*}
4537 {
4538 \bool_if:NT \g__enumext_minipage_active_vii_bool
4539 {
4540 __enumext_minipage:w [t] { \g__enumext_minipage_right_vii_dim }
4541 \legacy_if_gset_false:n { @minipage }
4542 \skip_vertical:N \c_zero_skip
4543 \par\addvspace { \g__enumext_minipage_right_skip }
4544 \bool_if:NF \g__enumext_minipage_center_vii_bool
4545 {
4546 \tl_put_left:Nn \g__enumext_miniright_code_vii_tl
4547 {
4548 \centering
4549 }
4550 }
4551 \vbox_set_top:Nn \l__enumext_miniright_code_vii_box
4552 {
4553 \tl_use:N \g__enumext_miniright_code_vii_tl
4554 }
4555 \box_use_drop:N \l__enumext_miniright_code_vii_box
4556 \skip_vertical:N \c_zero_skip
4557 __enumext_endminipage:
4558 \par\addvspace{ \g__enumext_minipage_after_skip }
4559 }
4560 \bool_gset_false:N \g__enumext_minipage_active_vii_bool
4561 \bool_gset_true:N \g__enumext_minipage_center_vii_bool
4562 \tl_gclear:N \g__enumext_miniright_code_vii_tl
4563 \dim_gzero:N \g__enumext_minipage_right_vii_dim

```

```

4564 \bool_gset_false:N \g__enumext_starred_bool
4565 }

```

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

`\__enumext_stop_mini_viii:`

```

4566 \cs_new_protected:Nn __enumext_start_mini_viii:
4567 {
4568 \dim_compare:nNt { \l__enumext_minipage_right_viii_dim } > { \c_zero_dim }
4569 {
4570 \dim_set:Nn \l__enumext_minipage_left_viii_dim
4571 {
4572 \linewidth
4573 - \l__enumext_minipage_right_viii_dim
4574 - \l__enumext_minipage_hsep_viii_dim
4575 }
4576 \bool_set_true:N \l__enumext_minipage_active_viii_bool
4577 \dim_gset_eq:NN
4578 \g__enumext_minipage_right_viii_dim
4579 \l__enumext_minipage_right_viii_dim
4580 __enumext_mini_addvspace_viii:
4581 \nointerlineskip\noindent
4582 __enumext_mini_page{ \l__enumext_minipage_left_viii_dim }
4583 }
4584 }
4585 \cs_new_protected:Nn __enumext_stop_mini_viii:
4586 {
4587 \bool_if:NTF \l__enumext_minipage_active_viii_bool
4588 {
4589 __enumext_stop_list:
4590 \IfDocumentMetadataTF { \tag_resume:n {keyans*} } { }
4591 \end__enumext_mini_page
4592 \hfill
4593 \bool_gset_true:N \g__enumext_minipage_active_viii_bool
4594 }
4595 {
4596 __enumext_stop_list:
4597 }
4598 }
4599 __enumext_after_env:n {keyans*}
4600 {
4601 \bool_if:NT \g__enumext_minipage_active_viii_bool
4602 {
4603 __enumext_mini_page{ \g__enumext_minipage_right_viii_dim }
4604 \par\addvspace { \g__enumext_minipage_right_skip }
4605 \bool_if:NF \g__enumext_minipage_center_viii_bool
4606 {
4607 \tl_put_left:Nn \g__enumext_miniright_code_viii_tl
4608 {
4609 \centering
4610 }
4611 }
4612 \vbox_set_top:Nn \l__enumext_miniright_code_viii_box
4613 {
4614 \tl_use:N \g__enumext_miniright_code_viii_tl
4615 }
4616 \box_use_drop:N \l__enumext_miniright_code_viii_box
4617 \end__enumext_mini_page
4618 \par\addvspace{ \g__enumext_minipage_after_skip }
4619 }
4620 \bool_gset_false:N \g__enumext_minipage_active_viii_bool
4621 \bool_gset_true:N \g__enumext_minipage_center_viii_bool
4622 \tl_gclear:N \g__enumext_miniright_code_viii_tl
4623 \dim_gzero:N \g__enumext_minipage_right_viii_dim
4624 }

```

(End of definition for `\__enumext_start_mini_viii:` and `\__enumext_stop_mini_viii:`)

### 13.44 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 `\__enumext_first_item_tmp_vii:` and next to `\item` equal to `\__enumext_start_item_tmp_vii:` which we will redefine later. Unlike the implementation used by the `shortlst`

package, we will not set the values of `\rightskip` and `\@rightskip` equal to `\@flushglue` whose value is `0.0pt plus 1.0 fil`, in the tests I have performed this fails in some circumstances and different results are obtained when using pdf $\TeX$  and Lua $\TeX$ .

```

4625 \NewDocumentEnvironment{enumext*}{ o }
4626 {
4627 __enumext_safe_exec_vii:
4628 __enumext_parse_keys_vii:n {#1}
4629 __enumext_before_list_vii:
4630 __enumext_start_store_level_vii:
4631 __enumext_start_list:nn { }
4632 {
4633 __enumext_list_arg_two_vii:
4634 __enumext_before_keys_exec_vii:
4635 }
4636 \IfDocumentMetadataTF { \tag_suspend:n {enumext*} } { }
4637 __enumext_starred_columns_set_vii:
4638 \item[] \scan_stop:
4639 \cs_set_eq:NN __enumext_stop_item_tmp_vii: __enumext_first_item_tmp_vii:
4640 \cs_set_eq:NN \item __enumext_start_item_tmp_vii:
4641 \ignorespaces
4642 }
4643 {
4644 \IfDocumentMetadataTF { \tag_struct_end:n {tag=text-unit} } { }
4645 __enumext_stop_item_tmp_vii:
4646 __enumext_remove_extra_parsep_vii:
4647 __enumext_after_list_vii:
4648 }

```

(End of definition for `enumext*`. This function is documented on page 5.)

`\__enumext_safe_exec_vii:`

We will first call the function `\__enumext_is_not_nested:` which sets `\g__enumext_starred_bool` to true if we are NOT nested within `enumext`, then call the function `\__enumext_internal_mini_page:` to create the environment `__enumext_mini_page`, 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.

```

4649 \cs_new_protected:Nn __enumext_safe_exec_vii:
4650 {
4651 __enumext_is_not_nested:
4652 __enumext_internal_mini_page:
4653 \int_incr:N \l__enumext_level_h_int
4654 \int_compare:nNnT { \l__enumext_level_h_int } > { 1 }
4655 {
4656 \msg_error:nn { enumext } { nested }
4657 }
4658 \int_compare:nNnT { \l__enumext_keyans_level_h_int } = { 1 }
4659 {
4660 \msg_error:nnn { enumext } { nested-horizontal } { keyans* }
4661 }
4662 \bool_set_true:N \l__enumext_starred_bool
4663 \bool_set_false:N \l__enumext_standar_bool
4664 __enumext_is_on_first_level:
4665 }

```

(End of definition for `\__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 `[⟨key = val⟩]` 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 `⟨keys⟩` to pass them to the storage *sequence* if the key `save-key` is not active.

```

4666 \cs_new_protected:Npn __enumext_parse_keys_vii:n #1
4667 {
4668 \tl_if_novalue:nF {#1}
4669 {
4670 \str_clear:N \l__enumext_series_str
4671 \keys_set:nn { enumext / enumext* } {#1}
4672 __enumext_parse_series:n {#1}
4673 __enumext_store_active_keys_vii:n {#1}
4674 }
4675 }

```

(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_vii:` and `\__enumext_start_mini_vii:` used by the keys `before*`, `mini-env`, `mini-right` and `mini-right*`.

```
4676 \cs_new_protected:Nn __enumext_before_list_vii:
4677 {
4678 __enumext_vspace_above_vii:
4679 __enumext_check_ans_active:
4680 __enumext_before_args_exec_vii:
4681 __enumext_start_mini_vii:
4682 }
```

(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:` which internally calls `\__enumext_stop_list:` and `\__enumext_stop_store_level_vii:` (§13.43.3) 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.

```
4683 \cs_new_protected:Nn __enumext_after_list_vii:
4684 {
4685 __enumext_stop_mini_vii:
4686 __enumext_after_stop_list_vii:
4687 __enumext_check_ans_key_hook:
4688 __enumext_vspace_below_vii:
4689 \bool_set_false:N \l__enumext_starred_bool
4690 __enumext_resume_save_counter:
4691 }
```

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

`\__enumext_start_store_level_vii:` and `\__enumext_stop_store_level_vii:` functions activate the “*storing structure*” mechanism in *sequence* for `\anskey` command and `anskey*` environment if `enumext*` are nested in `enumext`.

```
4692 \cs_new_protected:Nn __enumext_start_store_level_vii:
4693 {
4694 \bool_if:NT \l__enumext_store_active_bool
4695 {
4696 \int_compare:nNnT { \l__enumext_level_int } > { 0 }
4697 {
4698 __enumext_store_level_open_vii:
4699 }
4700 }
4701 }
4702 \cs_new_protected:Nn __enumext_stop_store_level_vii:
4703 {
4704 \bool_if:NT \l__enumext_store_active_bool
4705 {
4706 \int_compare:nNnT { \l__enumext_level_int } > { 0 }
4707 {
4708 __enumext_store_level_close_vii:
4709 }
4710 }
4711 }
```

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

### 13.44.1 The command `\item` in `enumext*`

`\__enumext_first_item_tmp_vii:` The `\__enumext_first_item_tmp_vii:` function will remove horizontal space equal to `\labelwidth` plus `\labelsep` to the left of the “*first*” `\item` in the environment at the point of execution of this function, where it is equal to the `\__enumext_stop_item_tmp_vii:` function inside the environment body definition.

```
4712 \cs_new_protected_nopar:Nn __enumext_first_item_tmp_vii:
4713 {
4714 \skip_horizontal:n
4715 {
4716 -\l__enumext_labelwidth_vii_dim - \l__enumext_labelsep_vii_dim
```



```

4717 }
4718 \ignorespaces
4719 }

```

(End of definition for `\__enumext_first_item_tmp_vii:`)

```

__enumext_start_item_tmp_vii:
__enumext_item_peek_args_vii:
__enumext_joined_item_vii:w
__enumext_standar_item_vii:w
__enumext_starred_item_vii:w
__enumext_starred_item_vii_aux_i:w
__enumext_starred_item_vii_aux_ii:w
__enumext_starred_item_vii_aux_iii:w

```

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`.

```

4720 \cs_new_protected_nopar:Nn __enumext_start_item_tmp_vii:
4721 {
4722 __enumext_stop_item_tmp_vii:
4723 \int_incr:N \l__enumext_item_column_pos_vii_int
4724 \int_gincr:N \g__enumext_item_count_all_vii_int
4725 __enumext_item_peek_args_vii:
4726 }

```

The function `\__enumext_item_peek_args_vii:` will handle the `\item(<number>)`. Look for the argument “(”, if it is present we will call the function `\__enumext_joined_item_vii:w (<number>)`, 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).

```

4727 \cs_new_protected:Nn __enumext_item_peek_args_vii:
4728 {
4729 \peek_meaning:NTF (
4730 { __enumext_joined_item_vii:w }
4731 { __enumext_joined_item_vii:w (1) }
4732 }

```

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`.

```

4733 \cs_new_protected:Npn __enumext_joined_item_vii:w (#1)
4734 {
4735 __enumext_starred_joined_item_vii:n {#1}
4736 \peek_meaning_remove:NTF *
4737 { __enumext_starred_item_vii:w }
4738 { __enumext_standar_item_vii:w }
4739 }

```

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[<custom>]` by means of the function `\__enumext_start_item_vii:w`, otherwise we will set the value of the variable `\l__enumext_wrap_label_vii_bool` handled by the `wrap-label` key to true and set the switch `\if@noitemarg` to true to execute the enumerated version of `\item` by means of the function `\__enumext_start_item_vii:w [ \l__enumext_label_vii_tl ]`.

```

4740 \cs_new_protected:Npn __enumext_standar_item_vii:w
4741 {
4742 \bool_set_false:N \l__enumext_item_starred_vii_bool
4743 \peek_meaning:NTF [
4744 {
4745 \bool_set_eq:NN \l__enumext_wrap_label_vii_bool \l__enumext_wrap_label_opt_vii_bool
4746 __enumext_start_item_vii:w
4747 }
4748 {
4749 \bool_set_true:N \l__enumext_wrap_label_vii_bool
4750 \legacy_if_set_true:n { @noitemarg }
4751 __enumext_start_item_vii:w [\l__enumext_label_vii_tl] \ignorespaces
4752 }
4753 }

```

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* [<symbol>]` and `\item* [<symbol>] [<offset>]`.

```

4754 \cs_new_protected:Npn __enumext_starred_item_vii:w
4755 {
4756 \bool_set_true:N \l__enumext_item_starred_vii_bool
4757 \bool_set_true:N \l__enumext_wrap_label_vii_bool
4758 \peek_meaning:NTF [
4759 { __enumext_starred_item_vii_aux_i:w }

```

```

4760 { __enumext_starred_item_vii_aux_ii:w }
4761 }
4762 \cs_new_protected:Npn __enumext_starred_item_vii_aux_i:w [#1]
4763 {
4764 \tl_gset:Nn \g__enumext_item_symbol_aux_vii_tl {#1}
4765 __enumext_starred_item_vii_aux_ii:w
4766 }
4767 \cs_new_protected:Npn __enumext_starred_item_vii_aux_iii:w
4768 {
4769 \peek_meaning:NTF [
4770 { __enumext_starred_item_vii_aux_iii:w }
4771 {
4772 \dim_set_eq:NN \l__enumext_item_symbol_sep_vii_dim \l__enumext_labelsep_vii_dim
4773 \legacy_if_set_true:n { @noitemarg }
4774 __enumext_start_item_vii:w [\l__enumext_label_vii_tl] \ignorespaces
4775 }
4776]
4777 \cs_new_protected:Npn __enumext_starred_item_vii_aux_iii:w [#1]
4778 {
4779 \dim_set:Nn \l__enumext_item_symbol_sep_vii_dim {#1}
4780 \legacy_if_set_true:n { @noitemarg }
4781 __enumext_start_item_vii:w [\l__enumext_label_vii_tl] \ignorespaces
4782 }

```

(End of definition for \\_\_enumext\_start\_item\_tmp\_vii: and others.)

\\_\_enumext\_fake\_make\_label\_vii:n

The \\_\_enumext\_fake\_make\_label\_vii:n function will be in charge of handling our definition of \item. First we increment the counter `enumXvii` for the enumerated items and activate support for the *check answers* mechanism, followed by support for `\item*[\langle symbol \rangle][\langle offset \rangle]` if present, then the `wrap-label` and `wrap-label*` keys which we execute using `\makebox` whose width will be given by the `labelwidth` key and position by the `align` key, inside the argument of this we will execute the `font` key together with the function defined by the `wrap-label` or `wrap-label*` keys. Finally we execute the `labelsep` key applying a `\skip_horizontal:N` and `\ignorespaces`.

- For compatibility with *tagged* PDF and *hyperref* when an environment `enumext` is nested in `enumext*` and the key `save-ans` is not active need setting the `\if@hyper@item` switch to “true”. The explanation for this is given by the master Heiko Oberdiek on `\refstepcounter{enumi}` twice (or more) creates destination with the same identifier. This patch is only needed if you are running `pdflatex` and not if you are running `lua1latex`

```

4783 \cs_new_protected_nopar:Npn __enumext_fake_make_label_vii:n #1
4784 {
4785 \legacy_if:nT { @noitemarg }
4786 {
4787 \legacy_if_set_false:n { @noitemarg }
4788 \legacy_if:nT { @nmbrrlist }
4789 {
4790 \IfDocumentMetadataTF
4791 {
4792 \bool_if:NT \l__enumext_hyperref_bool
4793 {
4794 \legacy_if_set_true:n { @hyper@item }
4795 }
4796 } { }
4797 \refstepcounter{enumXvii}
4798 \bool_if:NT \l__enumext_check_answers_bool
4799 {
4800 \int_gincr:N \g__enumext_item_number_int
4801 \bool_set_true:N \l__enumext_item_number_bool
4802 }
4803 }
4804 }
4805 \bool_if:NT \l__enumext_item_starred_vii_bool
4806 {
4807 \tl_if_blank:VT \g__enumext_item_symbol_aux_vii_tl
4808 {
4809 \tl_gset_eq:NN
4810 \g__enumext_item_symbol_aux_vii_tl \l__enumext_item_symbol_vii_tl
4811 }
4812 \mode_leave_vertical:
4813 \skip_horizontal:n { -\l__enumext_item_symbol_sep_vii_dim }
4814 \hbox_overlap_left:n { \g__enumext_item_symbol_aux_vii_tl }
4815 \skip_horizontal:N \l__enumext_item_symbol_sep_vii_dim

```

```

4816 \tl_gclear:N \g__enumext_item_symbol_aux_vii_tl
4817 }
4818 \makebox[\l__enumext_labelwidth_vii_dim][\l__enumext_align_label_vii_str]
4819 {
4820 \tl_use:N \l__enumext_label_font_style_vii_tl
4821 \bool_if:NTF \l__enumext_wrap_label_vii_bool
4822 {
4823 __enumext_wrapper_label_vii:n {#1}
4824 }
4825 { #1 }
4826 }
4827 \skip_horizontal:N \l__enumext_labelsep_vii_dim \ignorespaces
4828 }

```

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

### 13.44.2 Real definition of `\item` in `enumext*`

The functions `\__enumext_start_item_vii:w` and `\__enumext_stop_item_vii:` executing the true definition of `\item` inside the `enumext*` environment, unlike the implementation in `shortlst` we will NOT use an extra group and the plain form of the `lrbox` environment.

`\__enumext_start_item_vii:w` 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, after that we will start capturing `\item` and “*item content*” in a *horizontal box* where the width will be `\itemwidth` plus `\labelwidth` plus `\labelsep`.

```

4829 \cs_new_protected_nopar:Npn __enumext_start_item_vii:w [#1]
4830 {
4831 \cs_set_eq:NN __enumext_stop_item_tmp_vii: __enumext_stop_item_vii:
4832 \hbox_set_to_wd:Nnw \l__enumext_item_text_vii_box
4833 {
4834 \l__enumext_joined_width_vii_dim
4835 + \l__enumext_labelwidth_vii_dim
4836 + \l__enumext_labelsep_vii_dim
4837 }

```

Redefine the `\footnote` command.

```
4838 __enumext_renew_footnote_starred:
```

Now we insert our *sockets* for *tagging* PDF support and run `\item`.

```

4839 __enumext_start_list_tag:n {enumext*}
4840 __enumext_fake_make_label_vii:n {#1}
4841 __enumext_stop_start_list_tag:

```

Finally we open the `minipage` environment, capture the “*item content*”, make `\parindent` take the value of the key `listparindent` and `\parskip` take the value of the key `parsep`, then execute the keys `itemindent` and `first`.

Here the use of `\unskip` and `\skip_horizontal:n` with the value of `listparindent` is necessary, otherwise an unwanted space is created when using `\item[⟨opt⟩]` and the value passed to the key `itemindent` is incremented.

```

4842 __enumext_minipage:w [t]{ \l__enumext_joined_width_vii_dim }
4843 \dim_set_eq:NN \parindent \l__enumext_listparindent_vii_dim
4844 \skip_set_eq:NN \parskip \l__enumext_parsep_vii_skip
4845 __enumext_unskip_unkern:
4846 __enumext_unskip_unkern:
4847 \skip_horizontal:n { -\l__enumext_listparindent_vii_dim } \ignorespaces
4848 \tl_use:N \l__enumext_fake_item_indent_vii_tl
4849 \tl_use:N \l__enumext_after_list_args_vii_tl
4850 }

```

The `\__enumext_stop_item_vii:` function will finish the fetching `\item` and “*item content*” by closing the `minipage` environment, the *sockets* for *tagging* PDF and the *horizontal box*.

```

4851 \cs_new_protected_nopar:Nn __enumext_stop_item_vii:
4852 {
4853 __enumext_endminipage:
4854 __enumext_stop_list_tag:n {enumext*}
4855 \hbox_set_end:

```

Here we will reduce the *warnings* a bit by setting the value of `\hbadness` to `10000`, print `\item` and “*item content*” from the *horizontal box*.

```

4856 \int_set:Nn \hbadness { 10000 }
4857 \box_use_drop:N \l__enumext_item_text_vii_box

```

Finally apply the *vertical space* between rows set by `itemsep` key passed to `\parsep` using `\par\noindent` and *horizontal space* between columns set by `columns-sep` key using `\skip_horizontal:N`.

```

4858 \int_compare:nNnTF
4859 { \l__enumext_item_column_pos_vii_int } = { \l__enumext_columns_vii_int }
4860 {
4861 \par\noindent
4862 \int_zero:N \l__enumext_item_column_pos_vii_int
4863 }
4864 {
4865 \skip_horizontal:N \l__enumext_columns_sep_vii_dim
4866 }
4867 }

```

(End of definition for `\__enumext_start_item_vii:w` and `\__enumext_stop_item_vii:.`)

`\__enumext_remove_extra_parsep_vii:`

Remove the extra *vertical space* equal to `\parsep=\itemsep` when the total number of `\item` is divisible by the number of `\item` in the last row of the environment. Here the use of `\unskip` or `\removeatlastskip` fails and does not obtain the expected result, using `\vspace` is the option and in this case, we can use a simplified version since we are always in *(vertical mode)*.

```

4868 \cs_new_protected:Nn __enumext_remove_extra_parsep_vii:
4869 {
4870 \int_compare:nNnTF
4871 {
4872 \int_mod:nn
4873 { \g__enumext_item_count_all_vii_int } { \l__enumext_columns_vii_int }
4874 }
4875 =
4876 { 0 }
4877 {
4878 \para_end:
4879 \skip_vertical:n { -\l__enumext_itemsep_vii_skip }
4880 \skip_vertical:N \c_zero_skip
4881 \int_gzero:N \g__enumext_item_count_all_vii_int
4882 }
4883 }

```

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

As we don't want our check to be executed `check-ans` by levels but on the complete list, we will take it out of the `enumext*` environment using the “hook” function `\__enumext_after_env:nn`.

```

4884 __enumext_after_env:nn {enumext*}
4885 {
4886 __enumext_execute_after_env:
4887 }

```

### 13.45 The environment `keyans*`

`keyans*`

The implementation of `keyans*` environment is the similar as that used by the `enumext*` environment except for the `\__enumext_check_starred_cmd:n` function added in the second part.

```

4888 \NewDocumentEnvironment{keyans*}{o}
4889 {
4890 __enumext_safe_exec_viii:
4891 __enumext_parse_keys_viii:n {#1}
4892 __enumext_before_list_viii:
4893 __enumext_start_list:nn { }
4894 {
4895 __enumext_list_arg_two_viii:
4896 __enumext_before_keys_exec_viii:
4897 }
4898 \IfDocumentMetadataTF { \tag_suspend:n {keyans*} } { }
4899 __enumext_starred_columns_set_viii:
4900 \item[] \scan_stop:
4901 \cs_set_eq:NN __enumext_stop_item_tmp_viii: __enumext_first_item_tmp_viii:
4902 \cs_set_eq:NN \item __enumext_start_item_tmp_viii:
4903 \ignorespaces
4904 }
4905 {
4906 \IfDocumentMetadataTF { \tag_struct_end:n {tag=text-unit} } { }
4907 __enumext_stop_item_tmp_viii:
4908 __enumext_remove_extra_parsep_viii:
4909 __enumext_check_starred_cmd:n { item }

```

```

4910 __enumext_after_list_viii:
4911 }

```

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

`\__enumext_safe_exec_viii:`

The `\__enumext_safe_exec_viii:` function will first check if the `save-ans` key is active and only when this is true the environment will be available, it will increment the value of `\l__enumext_keyans_level_h_int` and return an error message when we are nesting the environment, then it will call the `\__enumext_keyans_name_and_start:` function in charge of saving the name of the environment and the line it is running on, then it will check if we are trying to nest `keyans*` in `enumext*` returning an error and we will set `\l__enumext_starred_bool` to true, finally we will check if we are within the appropriate level within the `enumext` environment.

```

4912 \cs_new_protected:Nn __enumext_safe_exec_viii:
4913 {
4914 \bool_if:NF \l__enumext_store_active_bool
4915 {
4916 \msg_error:nnnn { enumext } { wrong-place } { keyans* } { save-ans }
4917 }
4918 \int_incr:N \l__enumext_keyans_level_h_int
4919 \int_compare:nNnT { \l__enumext_keyans_level_h_int } > { 1 }
4920 {
4921 \msg_error:nn { enumext } { nested }
4922 }
4923 __enumext_keyans_name_and_start:
4924 \bool_if:NT \l__enumext_starred_bool
4925 {
4926 \msg_error:nnn { enumext } { nested-horizontal } { enumext* }
4927 }
4928 \bool_set_true:N \l__enumext_starred_bool
4929 % Set false for interfering with enumext nested in keyans* (yes, its possible and crayze)
4930 \bool_set_false:N \l__enumext_store_active_bool
4931 \int_compare:nNnT { \l__enumext_level_int } > { 1 }
4932 {
4933 \msg_error:nn { enumext } { keyans-wrong-level }
4934 }
4935 }

```

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

`\__enumext_parse_keys_viii:n`

Parse [`<key = val>`] for `keyans*`.

```

4936 \cs_new_protected:Npn __enumext_parse_keys_viii:n #1
4937 {
4938 \tl_if_novalue:nF {#1}
4939 {
4940 \keys_set:nn { enumext / keyans* } {#1}
4941 }
4942 }

```

(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 `{<code>}` defined by the `before*` key if it is active, the call the function `\__enumext_start_mini_viii:` handle by `mini-env`.

```

4943 \cs_new_protected:Nn __enumext_before_list_viii:
4944 {
4945 __enumext_vspace_above_viii:
4946 __enumext_before_args_exec_viii:
4947 __enumext_start_mini_viii:
4948 }

```

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

`\__enumext_after_list_viii:`

The function `\__enumext_after_list_viii:` first call the function `\__enumext_stop_mini_viii:`, then apply the `{<code>}` handled by the `after` key together with the *vertical space* handled by the `below` key if they are present.

```

4949 \cs_new_protected:Nn __enumext_after_list_viii:
4950 {
4951 __enumext_stop_mini_viii:
4952 __enumext_after_stop_list_viii:
4953 __enumext_vspace_below_viii:
4954 }

```

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

### 13.45.1 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* (`<number>`) which works in the same way as in the `enumext*` environment. In simple terms we want to store the `<label>` next to the `[<content>]` if it is present in the *sequence* and *prop list* defined by `save-ans` key for `\item*`, `\item*<content>`, `\item<number>*` and `\item<number>*<content>` commands.

`\__enumext_first_item_tmp_viii:`

The `\__enumext_first_item_tmp_viii:` function will remove horizontal space equal to `\labelwidth` plus `\labelsep` to the left of the “*first*” `\item` in the environment at the point of execution of this function, where it is equal to the `\__enumext_stop_item_tmp_viii:` function inside the environment body definition.

```

4955 \cs_new_protected_nopar:Nn __enumext_first_item_tmp_viii:
4956 {
4957 \skip_horizontal:n
4958 {
4959 -__enumext_labelwidth_viii_dim - __enumext_labelsep_viii_dim
4960 }
4961 \ignorespaces
4962 }

```

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

`\__enumext_start_item_tmp_viii:`

`\__enumext_item_peek_args_viii:`

`\__enumext_joined_item_viii:w`

`\__enumext_standar_item_viii:w`

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`.

```

4963 \cs_new_protected_nopar:Nn __enumext_start_item_tmp_viii:
4964 {
4965 __enumext_stop_item_tmp_viii:
4966 \int_incr:N \l__enumext_item_column_pos_viii_int
4967 \int_gincr:N \g__enumext_item_count_all_viii_int
4968 __enumext_item_peek_args_viii:
4969 }

```

The function `\__enumext_item_peek_args_viii:` will handle the `\item<number>`. Look for the argument “`(`”, if it is present we will call the function `\__enumext_joined_item_viii:w (<number>)`, 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)`.

```

4970 \cs_new_protected:Nn __enumext_item_peek_args_viii:
4971 {
4972 \peek_meaning:NTF (
4973 { __enumext_joined_item_viii:w }
4974 { __enumext_joined_item_viii:w (1) }
4975 }

```

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`.

```

4976 \cs_new_protected:Npn __enumext_joined_item_viii:w (#1)
4977 {
4978 __enumext_starred_joined_item_viii:n {#1}
4979 \peek_meaning_remove:NTF *
4980 { __enumext_starred_item_viii:w }
4981 { __enumext_standar_item_viii:w }
4982 }

```

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[<custom>]` 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@notitemarg` to true to execute the enumerated version of `\item` by means of the function `\__enumext_start_item_viii:w [ \l__enumext_label_viii_tl ]`.

```

4983 \cs_new_protected:Npn __enumext_standar_item_viii:w
4984 {
4985 \bool_set_false:N \l__enumext_item_starred_viii_bool
4986 \int_zero:N \l__enumext_item_star_exec_int
4987 \peek_meaning:NTF [
4988 {
4989 \bool_set_eq:NN \l__enumext_wrap_label_viii_bool \l__enumext_wrap_label_opt_viii_bool

```

```

4990 __enumext_start_item_viii:w
4991 }
4992 {
4993 \bool_set_true:N __enumext_wrap_label_viii_bool
4994 \legacy_if_set_true:n { @noitemarg }
4995 __enumext_start_item_viii:w [__enumext_label_viii_tl] \ignorespaces
4996 }
4997 }

```

(End of definition for \\_\_enumext\_start\_item\_tmp\_viii: and others.)

```

__enumext_starred_item_viii:w
__enumext_starred_item_viii_aux_i:w
__enumext_starred_item_viii_aux_ii:w
__enumext_starred_item_exec:

```

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]`.

```

4998 \cs_new_protected:Npn __enumext_starred_item_viii:w
4999 {
5000 \bool_set_true:N __enumext_item_starred_viii_bool
5001 \int_incr:N __enumext_item_star_exec_int
5002 \bool_set_true:N __enumext_wrap_label_viii_bool
5003 \peek_meaning:NTF [
5004 { __enumext_starred_item_viii_aux_i:w }
5005 { __enumext_starred_item_viii_aux_ii:w }
5006 }

```

The function \\_\_enumext\_starred\_item\_viii\_aux\_i:w will save the *optional argument* to `\item*` in `\__enumext_store_current_opt_arg_tl` and will save this argument along with the spacing set by the key `save-sep` in variable `\__enumext_store_current_label_tl` if present, then call the function `\__enumext_starred_item_viii_aux_ii:w`.

```

5007 \cs_new_protected:Npn __enumext_starred_item_viii_aux_i:w [#1]
5008 {
5009 \tl_clear:N __enumext_store_current_label_tl
5010 \tl_if_novalue:nF { #1 }
5011 {
5012 \tl_if_empty:NF __enumext_store_keyans_item_opt_sep_tl
5013 {
5014 \tl_put_right:Ne __enumext_store_current_label_tl
5015 {
5016 __enumext_store_keyans_item_opt_sep_tl
5017 }
5018 \tl_put_right:Ne __enumext_store_current_label_tl { #1 }
5019 }
5020 \tl_set:Ne __enumext_store_current_opt_arg_tl { #1 }
5021 }
5022 __enumext_starred_item_viii_aux_ii:w
5023 }
5024 \cs_new_protected:Npn __enumext_starred_item_viii_aux_ii:w
5025 {
5026 \legacy_if_set_true:n { @noitemarg }
5027 __enumext_start_item_viii:w [__enumext_label_viii_tl] \ignorespaces
5028 }

```

The function \\_\_enumext\_starred\_item\_exec: will be in charge of storing the current *label* for `\item*` followed by the `[\langle content \rangle]` for `\item*[\langle content \rangle]` if present in the *sequence* and *prop list* set by the `save-ans` key. In this same function the keys `show-ans`, `show-pos` and `save-ref` are implemented.

```

5029 \cs_new_protected:Nn __enumext_starred_item_exec:
5030 {
5031 \tl_put_left:Ne __enumext_store_current_label_tl { __enumext_label_viii_tl }
5032 __enumext_store_addto_prop:V __enumext_store_current_label_tl
5033 __enumext_keyans_store_ref:
5034 \tl_put_left:Ne __enumext_store_current_label_tl { \item }
5035 __enumext_keyans_addto_seq_link:
5036 \int_gincr:N __enumext_check_starred_cmd_int
5037 \bool_if:NT __enumext_show_answer_bool
5038 {
5039 __enumext_print_keyans_box:NN __enumext_labelwidth_i_dim __enumext_labelsep_i_dim
5040 }
5041 \bool_if:NT __enumext_show_position_bool
5042 {
5043 \tl_set:Ne __enumext_mark_answer_sym_tl
5044 {
5045 \group_begin:
5046 \exp_not:N \normalfont

```



```

5047 \exp_not:N \footnotesize [\int_eval:n
5048 {
5049 \prop_count:c { g__enumext_ \l__enumext_store_name_tl _prop }
5050 }
5051]
5052 \group_end:
5053 }
5054 __enumext_print_keyans_box:NN \l__enumext_labelwidth_i_dim \l__enumext_labelsep_i_dim
5055 }
5056 }

```

(End of definition for `\__enumext_starred_item_viii:w` and others.)

`\__enumext_fake_make_label_viii:n`

The implementation at this is very similar to that of the `enumext*` environment.

```

5057 \cs_new_protected:Npn __enumext_keyans_wrapper_label_viii:n #1
5058 {
5059 \bool_lazy_all:nT
5060 {
5061 { \bool_if_p:N \l__enumext_wrap_label_viii_bool }
5062 { \bool_if_p:N \l__enumext_show_answer_bool }
5063 { \int_compare_p:nNn { \l__enumext_item_star_exec_int } = { 1 } }
5064 { \cs_if_exist_p:N __enumext_keyans_wrapper_item:n }
5065 }
5066 {
5067 \cs_set_eq:NN __enumext_wrapper_label_viii:n __enumext_keyans_wrapper_item:n
5068 }
5069 \bool_if:NTF \l__enumext_wrap_label_viii_bool
5070 {
5071 __enumext_wrapper_label_viii:n {#1}
5072 }
5073 { #1 }
5074 }
5075 \cs_new_protected_nopar:Npn __enumext_fake_make_label_viii:n #1
5076 {
5077 \legacy_if:nT { @noitemarg }
5078 {
5079 \legacy_if_set_false:n { @noitemarg }
5080 \legacy_if:nT { @nmbrrlist }
5081 {
5082 \refstepcounter{enumXviii}
5083 }
5084 }
5085 \bool_if:NT \l__enumext_item_starred_viii_bool
5086 {
5087 __enumext_starred_item_exec:
5088 }
5089 \makebox[\l__enumext_labelwidth_viii_dim][\l__enumext_align_label_viii_str]
5090 {
5091 \tl_use:N \l__enumext_label_font_style_viii_tl
5092 __enumext_keyans_wrapper_label_viii:n {#1}
5093 %%\bool_if:NTF \l__enumext_wrap_label_viii_bool
5094 %% {
5095 %% __enumext_wrapper_label_viii:n {#1}
5096 %% }
5097 %% { #1 }
5098 }
5099 \skip_horizontal:N \l__enumext_labelsep_viii_dim \ignorespaces
5100 }

```

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

### 13.45.2 Real definition of `\item` in `keyans*`

`\__enumext_start_item_viii:w`

The implementation at this is very similar to that of the `enumext*` environment.

`\__enumext_stop_item_viii:`

```

5101 \cs_new_protected_nopar:Npn __enumext_start_item_viii:w [#1]
5102 {
5103 \cs_set_eq:NN __enumext_stop_item_tmp_viii: __enumext_stop_item_viii:
5104 \hbox_set_to_wd:Nnw \l__enumext_item_text_viii_box
5105 {
5106 \l__enumext_joined_width_viii_dim
5107 + \l__enumext_labelwidth_viii_dim
5108 + \l__enumext_labelsep_viii_dim

```

```

5109 }
5110 __enumext_renew_footnote_starred:
5111 __enumext_start_list_tag:n {keyans*}
5112 __enumext_fake_make_label_viii:n {#1}
5113 __enumext_stop_start_list_tag:
5114 __enumext_minipage:w [t]{ __enumext_joined_width_viii_dim }
5115 \dim_set_eq:NN \parindent __enumext_listparindent_viii_dim
5116 \skip_set_eq:NN \parskip __enumext_parsep_viii_skip
5117 __enumext_unskip_unkern:
5118 __enumext_unskip_unkern:
5119 \skip_horizontal:n { -__enumext_listparindent_viii_dim } \ignorespaces
5120 \tl_use:N __enumext_fake_item_indent_viii_tl
5121 \bool_if:NT __enumext_item_starred_viii_bool
5122 {
5123 __enumext_keyans_show_item_opt:
5124 }
5125 \tl_use:N __enumext_after_list_args_viii_tl
5126 }
5127 \cs_new_protected_nopar:Nn __enumext_stop_item_viii:
5128 {
5129 __enumext_endminipage:
5130 __enumext_stop_list_tag:n {keyans*}
5131 \hbox_set_end:
5132 \int_set:Nn \hbadness { 10000 }
5133 \box_use_drop:N __enumext_item_text_viii_box
5134 \int_compare:nNnTF
5135 { __enumext_item_column_pos_viii_int } = { __enumext_columns_viii_int }
5136 {
5137 \par\noindent
5138 \int_zero:N __enumext_item_column_pos_viii_int
5139 }
5140 {
5141 \skip_horizontal:N __enumext_columns_sep_viii_dim
5142 }
5143 }

```

(End of definition for \\_\_enumext\_start\_item\_viii:w and \\_\_enumext\_stop\_item\_viii:.)

\\_\_enumext\_remove\_extra\_parsep\_viii:

The implementation at this is very similar to that of the `enumext*` environment.

```

5144 \cs_new_protected:Nn __enumext_remove_extra_parsep_viii:
5145 {
5146 \int_compare:nNnT
5147 {
5148 \int_mod:nn
5149 { \g__enumext_item_count_all_viii_int }
5150 { __enumext_columns_viii_int }
5151 }
5152 =
5153 { 0 }
5154 {
5155 \para_end:
5156 \skip_vertical:n { -__enumext_itemsep_viii_skip }
5157 \skip_vertical:N \c_zero_skip
5158 \int_gzero:N \g__enumext_item_count_all_viii_int
5159 }
5160 }

```

(End of definition for \\_\_enumext\_remove\_extra\_parsep\_viii:.)

### 13.46 The command \getkeyans

\getkeyans  
\\_\_enumext\_getkeyans\_aux:n  
\\_\_enumext\_getkeyans:n

The `\getkeyans` command takes a *mandatory argument* of the form  $\langle \textit{store name} : \textit{position} \rangle$ . Retrieve a “single content” stored by `\anskey`, `\anspic*` and `\item*` and `anskey*` from *prop list* defined by `save-anskey`.

```

5161 \NewDocumentCommand \getkeyans { m }
5162 {
5163 \exp_args:Ne __enumext_getkeyans_aux:n
5164 { \tl_to_str:e { \text_expand:n {#1} } }
5165 }

```

The internal function `\__enumext_getkeyans_aux:n` is in charge of *splitting* the *mandatory argument* using “.”. If “.” is omitted it will return an error.

```

5166 \cs_new_protected:Npn __enumext_getkeyans_aux:n #1
5167 {
5168 \str_if_in:nnTF {#1} { : }
5169 {
5170 \use:e
5171 {
5172 \cs_set:Npn \exp_not:N __enumext_tmp:w ##1 \c_colon_str ##2 \scan_stop:
5173 { {##1} {##2} }
5174 }
5175 \exp_after:wN __enumext_getkeyans:nn __enumext_tmp:w #1 \scan_stop:
5176 }
5177 { \msg_error:nnn { enumext } { missing-colon } {#1} }
5178 }

```

The internal function `\__enumext_getkeyans:nn` will check for the existence of the *prop list*, if it does not exist it will return an error message, then it will fetch the content specified by the *second argument* from *prop list*.

```

5179 \cs_new_protected:Npn __enumext_getkeyans:nn #1 #2
5180 {
5181 \prop_if_exist:cTF { g__enumext_#1_prop }
5182 {
5183 \prop_item:cn { g__enumext_#1_prop } {#2}
5184 }
5185 {
5186 \msg_error:nnn { enumext } { undefined-storage-anskey } {#1}
5187 }
5188 }

```

(End of definition for `\getkeyans`, `\__enumext_getkeyans_aux:n`, and `\__enumext_getkeyans:nn`. This function is documented on page 17.)

### 13.47 The command `\printkeyans`

The `\printkeyans` command prints “*all stored content*” in the *sequence* defined by the `save-ans` key. The first thing we will do is define a set of (*filtered keys*) 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 (*keys*) for `\printkeyans*` and will be set by `\setenumext[⟨print*⟩]` and the variable `\l__enumext_print_keyans_vii_tl` will have the default keys for the environment `enumext*` nested within the *sequence* and will be set by `\setenumext[⟨print,*⟩]`, the rest of the variables will be for the environment `enumext` and will be set by `\setenumext[⟨print,level⟩]`.

```

5189 \keys_define:nn { enumext / print }
5190 {
5191 print* .code:n = \keys_precompile:neN { enumext / enumext* }
5192 { __enumext_filter_save_key:n {#1} }
5193 \l__enumext_print_keyans_starred_tl, % starred cmd
5194 print* .initial:n = { nosep, label=\arabic*., columns=2, first=\small, font=\small },
5195 print-1 .code:n = \keys_precompile:neN { enumext / level-1 }
5196 { __enumext_filter_save_key:n {#1} }
5197 \l__enumext_print_keyans_i_tl,
5198 print-1 .initial:n = { nosep, label=\arabic*., columns=2, first=\small, font=\small },
5199 print-2 .code:n = \keys_precompile:neN { enumext / level-2 }
5200 { __enumext_filter_save_key:n {#1} }
5201 \l__enumext_print_keyans_ii_tl,
5202 print-2 .initial:n = { nosep, label=(\alph*), first=\small, font=\small },
5203 print-3 .code:n = \keys_precompile:neN { enumext / level-3 }
5204 { __enumext_filter_save_key:n {#1} }
5205 \l__enumext_print_keyans_iii_tl,
5206 print-3 .initial:n = { nosep, label=\roman*., first=\small, font=\small },
5207 print-4 .code:n = \keys_precompile:neN { enumext / level-4 }
5208 { __enumext_filter_save_key:n {#1} }
5209 \l__enumext_print_keyans_iv_tl,
5210 print-4 .initial:n = { nosep, label=\Alph*., first=\small, font=\small },
5211 print-* .code:n = \keys_precompile:neN { enumext / enumext* }
5212 { __enumext_filter_save_key:n {#1} }
5213 \l__enumext_print_keyans_vii_tl, % starred nested
5214 print-* .initial:n = { nosep, label=\arabic*., first=\small, font=\small },
5215 }

```

- The reason for storing *(keys)* 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`

`\__enumext_printkeyans:nnn`

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`.

```

5216 \NewDocumentCommand \printkeyans { s O{} m }
5217 {
5218 \group_begin:
5219 \tl_use:N __enumext_print_keyans_i_tl
5220 \tl_use:N __enumext_print_keyans_ii_tl
5221 \tl_use:N __enumext_print_keyans_iii_tl
5222 \tl_use:N __enumext_print_keyans_iv_tl
5223 \tl_use:N __enumext_print_keyans_vii_tl
5224 __enumext_printkeyans:nnn { #1 } { #2 } { #3 }
5225 \group_end:
5226 }

```

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.

```

5227 \cs_new_protected:Npn __enumext_printkeyans:nnn #1 #2 #3
5228 {
5229 \seq_if_exist:cTF { g__enumext_#3_seq }
5230 {
5231 \seq_if_empty:cF { g__enumext_#3_seq }
5232 {

```

If the *starred argument* ‘\*’ is present we will check that the environment `enumext*` is not saved in the *sequence*, then execute the variable `\__enumext_print_keyans_starred_tl` that contains the default *(keys)* for the environment `enumext*`, we set `\__enumext_base_line_fix_bool` and `\__enumext_print_keyans_star_bool` to true for *baseline correction*, open the `enumext*` environment passing the *optional argument* and map the *sequence*, then set `\__enumext_base_line_fix_bool` and `\__enumext_print_keyans_star_bool` to false.

```

5233 \bool_if:nTF {#1}
5234 {
5235 \seq_if_in:cnTF { g__enumext_#3_seq } { \end{enumext*} }
5236 {
5237 \msg_error:nnnn { enumext } { print-starred } {#3} { enumext* }
5238 }
5239 {
5240 \tl_use:N __enumext_print_keyans_starred_tl
5241 \bool_set_true:N __enumext_base_line_fix_bool
5242 \bool_set_true:N __enumext_print_keyans_star_bool
5243 \begin{enumext*}[#2]
5244 \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
5245 \end{enumext*}
5246 \bool_set_false:N __enumext_base_line_fix_bool
5247 \bool_set_false:N __enumext_print_keyans_star_bool
5248 }
5249 }

```

Otherwise it will open the environment `enumext` passing the *optional argument* to the “first level” then map the *sequence*.

```

5250 {
5251 \begin{enumext}[#2]
5252 \seq_map_inline:cn { g__enumext_#3_seq } { ##1 }
5253 \end{enumext}
5254 }
5255 }
5256 }
5257 {
5258 \msg_error:nnn { enumext } { undefined-storage-anskey } {#3}
5259 }
5260 }

```

(End of definition for `\printkeyans` and `\__enumext_printkeyans:nnn`. This function is documented on page 18.)

### 13.48 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.

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`.

```
__enumext_filter_first_level:n
__enumext_filter_first_level_key:n
__enumext_filter_first_level_pair:nn
```

```
5261 \cs_new:Npn __enumext_filter_first_level:n #1
5262 {
5263 \use:e
5264 {
5265 \keyval_parse:NNn
5266 __enumext_filter_first_level_key:n
5267 __enumext_filter_first_level_pair:nn {#1}
5268 }
5269 }
```

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*`.

```
5270 \cs_new:Npn __enumext_filter_first_level_key:n #1
5271 {
5272 \str_case:nnF {#1}
5273 {
5274 { resume } {}
5275 { resume* } {}
5276 }
5277 { , { \exp_not:n {#1} } }
5278 }
```

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.

```
5279 \cs_new:Npn __enumext_filter_first_level_pair:nn #1#2
5280 {
5281 \str_case:nnF {#1}
5282 {
5283 { series } {}
5284 { resume } {}
5285 { save-ans } {}
5286 }
5287 { , { \exp_not:n {#1} } = { \exp_not:n {#2} } }
5288 }
```

(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`.

```
5289 \keys_define:nn { enumext / meta-families }
5290 {
5291 enumext-1 .code:n =
5292 {
5293 \keys_set:ne { enumext / level-1 }
5294 {
5295 __enumext_filter_first_level:n {#1}
5296 }
5297 } ,
5298 enumext-2 .code:n = { \keys_set:nn { enumext / level-2 } {#1} } ,
5299 enumext-3 .code:n = { \keys_set:nn { enumext / level-3 } {#1} } ,
5300 enumext-4 .code:n = { \keys_set:nn { enumext / level-4 } {#1} } ,
5301 keyans .code:n = { \keys_set:nn { enumext / keyans } {#1} } ,
5302 enumext* .code:n =
5303 {
5304 \keys_set:ne { enumext / enumext* }
5305 {
5306 __enumext_filter_first_level:n {#1}
5307 }
5308 } ,
5309 keyans* .code:n = { \keys_set:nn { enumext / keyans* } {#1} } ,
5310 print* .code:n = { \keys_set:nn { enumext / print } { print* = {#1} } } ,
5311 print-1 .code:n = { \keys_set:nn { enumext / print } { print-1 = {#1} } } ,
5312 print-2 .code:n = { \keys_set:nn { enumext / print } { print-2 = {#1} } } ,
5313 print-3 .code:n = { \keys_set:nn { enumext / print } { print-3 = {#1} } } ,
5314 print-4 .code:n = { \keys_set:nn { enumext / print } { print-4 = {#1} } } ,
```

```

5315 print-* .code:n = { \keys_set:nn { enumext / print } { print-* = {#1} } } ,
5316 unknown .code:n = { \msg_error:nn { enumext } { unknown-key-family } } ,
5317 }

```

We store them in the constant sequence `\c__enumext_all_families_seq` separated by commas.

```

5318 \seq_const_from_clist:Nn \c__enumext_all_families_seq
5319 {
5320 enumext-1, enumext-2, enumext-3, enumext-4, keyans, enumext*,
5321 keyans*, print-1, print-2, print-3, print-4, print-*, print*,
5322 }

```

`\setenumext` Now we define the user command `\setenumext`.

```

__enumext_set_parse:n
__enumext_set_error:nn
5323 \NewDocumentCommand \setenumext { 0{enumext,1} +m }
5324 {
5325 \seq_clear:N \l__enumext_setkey_tmpa_seq
5326 \seq_set_from_clist:Nn \l__enumext_setkey_tmpb_seq {#1}
5327 \int_set:Nn \l__enumext_setkey_tmpa_int
5328 {
5329 \seq_count:N \l__enumext_setkey_tmpb_seq
5330 }
5331 \int_compare:nNnTF { \l__enumext_setkey_tmpa_int } > { 1 }
5332 {
5333 \seq_pop_left:NN \l__enumext_setkey_tmpb_seq \l__enumext_setkey_tmpa_tl
5334 \seq_map_function:NN \l__enumext_setkey_tmpb_seq __enumext_set_parse:n
5335 \seq_set_map_e:NNn \l__enumext_setkey_tmpa_seq \l__enumext_setkey_tmpa_seq
5336 {
5337 \tl_use:N \l__enumext_setkey_tmpa_tl - ##1
5338 }
5339 }
5340 {
5341 \seq_put_right:Ne \l__enumext_setkey_tmpa_seq { \tl_trim_spaces:n {#1} }
5342 }
5343 \seq_if_empty:NNTF \l__enumext_setkey_tmpa_seq
5344 { \seq_map_inline:Nn \c__enumext_all_families_seq }
5345 { \seq_map_inline:Nn \l__enumext_setkey_tmpa_seq }
5346 {
5347 \keys_set:nn { enumext / meta-families } { ##1 = {#2} }
5348 }
5349 }

```

Internal functions used by the `\setenumext` command.

```

5350 \cs_new_protected:Npn __enumext_set_parse:n #1
5351 {
5352 \tl_set:Ne \l__enumext_setkey_tmpb_tl { \tl_trim_spaces:n {#1} }
5353 \clist_map_inline:nn { 0, 1, 2, 3, 4, * } % <- max level
5354 { \tl_remove_all:Nn \l__enumext_setkey_tmpb_tl {##1} }
5355 \tl_if_empty:NNTF \l__enumext_setkey_tmpb_tl
5356 {
5357 \seq_put_right:Ne \l__enumext_setkey_tmpa_seq
5358 { \tl_trim_spaces:n {#1} }
5359 }
5360 { __enumext_set_error:nn {#1} { } }
5361 }
5362 \cs_new_protected:Npn __enumext_set_error:nn #1 #2
5363 { \msg_error:nnn { enumext } { invalid-key } {#1} {#2} }

```

(End of definition for `\setenumext`, `\__enumext_set_parse:n`, and `\__enumext_set_error:nn`. This function is documented on page 6.)

### 13.49 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` First we will create a prop list `\c__enumext_meta_paths_prop` to handle the *optional argument*.

```

\c__enumext_meta_paths_prop
__enumext_add_meta_key:nnn
__enumext_def_meta_key:nnn
__enumext_def_meta_key:Vnn
5364 \prop_const_from_keyval:Nn \c__enumext_meta_paths_prop
5365 {
5366 {enumext,1} = level-1,
5367 {enumext,2} = level-2,
5368 {enumext,3} = level-3,
5369 {enumext,4} = level-4,

```

```

5370 {enumext*} = enumext*
5371 }

```

Now we create the user command taking care that unknown cannot be passed as an argument.

```

5372 \NewDocumentCommand \setenumextmeta { s O{enumext,1} m +m }
5373 {
5374 \str_if_eq:eeTF { \tl_trim_spaces:n {#3} } { unknown }
5375 { \msg_error:nn { enumext } { prohibited-unknown } }
5376 {
5377 \bool_if:nTF {#1}
5378 {
5379 \int_step_inline:nn { 4 }
5380 { __enumext_add_meta_key:nnn { enumext, ##1 } {#3} {#4} }
5381 __enumext_add_meta_key:nnn { enumext* } {#3} {#4}
5382 }
5383 { __enumext_add_meta_key:nnn {#2} {#3} {#4} }
5384 }
5385 }

```

The internal functions `\__enumext_add_meta_key:nnn` and `\__enumext_def_meta_key:nnn` will check the *optional argument* and create the “*meta-key*”.

```

5386 \cs_new_protected:Npn __enumext_add_meta_key:nnn #1
5387 {
5388 \tl_set:Nn \l__enumext_meta_path_tl {#1}
5389 \tl_replace_all:Nnn \l__enumext_meta_path_tl { ~ } {}
5390 \prop_get:NVNTF
5391 \c__enumext_meta_paths_prop \l__enumext_meta_path_tl \l__enumext_meta_path_tl
5392 { __enumext_def_meta_key:Vnn \l__enumext_meta_path_tl }
5393 {
5394 \msg_error:nnn { enumext } { unknown-set } {#1}
5395 \use_none:nn
5396 }
5397 }
5398 \cs_new_protected:Npn __enumext_def_meta_key:nnn #1#2#3
5399 {
5400 \bool_lazy_or:nnTF
5401 { \keys_if_exist:p:nn { enumext / #1 } {#2} }
5402 { \keys_if_exist:p:nn { enumext / enumext* } {#2} }
5403 { \msg_error:nnn { enumext } { already-defined } {#2} }
5404 {
5405 \keys_define:nn { enumext / #1 }
5406 {
5407 #2 .meta:n = {#3},
5408 #2 .value_forbidden:n = true
5409 }
5410 }
5411 }
5412 \cs_generate_variant:Nn __enumext_def_meta_key:nnn { V }

```

(End of definition for `\setenumextmeta` and others. This function is documented on page 6.)

### 13.50 The command `\foreachkeyans`

The command `\foreachkeyans` will execute a *loop* over the *prop list* and return its contents. The implementation code is adapted from the answer provided by Enrico Gregorio (@egreg) in [Expand a .cs defined by key inside the function](#).

`\foreachkeyans`

```

__enumext_parse_foreach_keys:nn
__enumext_parse_foreach_keys:n
__enumext_foreach_keyans:nn
__enumext_foreach_add_body:n

```

We define a set of *⟨keys⟩* for command and we will save the default values of these in `\g__enumext_foreach_default_keys_tl` to avoid the use of group.

```

5413 \keys_define:nn { enumext / foreach }
5414 {
5415 before .tl_set:N = \l__enumext_foreach_before_tl,
5416 before .value_required:n = true,
5417 after .tl_set:N = \l__enumext_foreach_after_tl,
5418 after .value_required:n = true,
5419 start .int_set:N = \l__enumext_foreach_start_int,
5420 start .value_required:n = true,
5421 stop .int_set:N = \l__enumext_foreach_stop_int,
5422 stop .value_required:n = true,
5423 step .int_set:N = \l__enumext_foreach_step_int,
5424 step .value_required:n = true,
5425 wrapper .cs_set_protected:Np = __enumext_foreach_wrapper:n #1,
5426 wrapper .value_required:n = true,

```



```

5427 sep .tl_set:N = \l__enumext_foreach_sep_tl,
5428 sep .value_required:n = true,
5429 unknown .code:n = { \l__enumext_parse_foreach_keys:n {#1} }
5430 }
5431 \keys_precompile:nnN { enumext / foreach }
5432 {
5433 before={},after={},start=1,step=1,stop=0,wrapper=#1,sep={; }
5434 }
5435 \g__enumext_foreach_default_keys_tl

```

Functions for handling unknown  $\langle keys \rangle$ .

```

5436 \cs_new_protected:Npn \l__enumext_parse_foreach_keys:nn #1#2
5437 {
5438 \tl_if_blank:nTF {#2}
5439 {
5440 \msg_error:nnn { enumext } { for-key-unknown } {#1}
5441 }
5442 {
5443 \msg_error:nnnn { enumext } { for-key-value-unknown } {#1} {#2}
5444 }
5445 }
5446 \cs_new_protected:Npn \l__enumext_parse_foreach_keys:n #1
5447 {
5448 \exp_args:NV \l__enumext_parse_foreach_keys:nn \l_keys_key_str {#1}
5449 }

```

We create the command.

```

5450 \NewDocumentCommand \foreachkeyans { +0{ } m }
5451 {
5452 \l__enumext_foreach_keyans:nn {#1} {#2}
5453 }

```

Finally the internal functions  $\l__enumext_foreach_keyans:nn$  and  $\l__enumext_foreach_add_body:n$  will loop through the prop list and print the contents.

```

5454 \cs_new_protected:Npn \l__enumext_foreach_keyans:nn #1 #2
5455 {
5456 \tl_use:N \g__enumext_foreach_default_keys_tl
5457 \keys_set:nn { enumext / foreach } {#1}
5458 \tl_set:Nn \l__enumext_foreach_name_prop_tl {#2}
5459 \prop_if_exist:cF { g__enumext_#2_prop }
5460 {
5461 \msg_error:nnn { enumext } { undefined-storage-anskey } {#2}
5462 }
5463 \int_compare:nNnT { \l__enumext_foreach_stop_int } = { 0 }
5464 {
5465 \int_set:Nn \l__enumext_foreach_stop_int
5466 { \prop_count:c { g__enumext_#2_prop } }
5467 }
5468 \seq_clear:N \l__enumext_foreach_print_seq
5469 \int_step_function:nnnN
5470 { \l__enumext_foreach_start_int }
5471 { \l__enumext_foreach_step_int }
5472 { \l__enumext_foreach_stop_int }
5473 \l__enumext_foreach_add_body:n
5474 \seq_use:NV \l__enumext_foreach_print_seq \l__enumext_foreach_sep_tl
5475 }
5476 \cs_new_protected:Npn \l__enumext_foreach_add_body:n #1
5477 {
5478 \seq_put_right:Ne \l__enumext_foreach_print_seq
5479 {
5480 \exp_not:V \l__enumext_foreach_before_tl
5481 \l__enumext_foreach_wrapper:n
5482 {
5483 \prop_item:cn { g__enumext_ \l__enumext_foreach_name_prop_tl _prop }{#1}
5484 }
5485 \exp_not:V \l__enumext_foreach_after_tl
5486 }
5487 }

```

(End of definition for  $\lforeachkeyans$  and others. This function is documented on page 17.)

### 13.51 Messages

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

```

5488 \msg_new:nnn { enumext } { package-load }
5489 {
5490 The ~ '#1' ~ package ~ is ~ already ~ loaded.
5491 }
5492 \msg_new:nnn { enumext } { package-not-load }
5493 {
5494 The ~ '#1' ~ package ~ will ~ be ~ loaded ~ as ~ a ~ dependency.
5495 }
5496 \msg_new:nnn { enumext } { package-load-foot }
5497 {
5498 The ~ '#1' ~ package ~ is ~ loaded ~ with ~ the ~ option ~ '#2'.
5499 }

```

Message used in the creation of counters by **enumext** package.

```

5500 \msg_new:nnn { enumext } { counters }
5501 {
5502 The ~ counter ~ '#1' ~ is ~ already ~ defined ~ by ~ some ~ \\
5503 package ~ or ~ macro, ~ it ~ cannot ~ be ~ continued.
5504 }

```

Message used by **align** and **mark-pos** keys.

```

5505 \msg_new:nnn { enumext } { unknown-choice }
5506 {
5507 The ~ value ~ '#3' ~ for ~ '#1' ~ key ~ is ~ invalid ~ use ~ ('#2').
5508 }

```

Message used by reserved **anskey\*** environment by **enumext** package.

```

5509 \msg_new:nnnn { enumext } { anskey-env-error }
5510 {
5511 The ~ '#1' ~ environment ~is~ reserved ~ by ~\\
5512 'enumext' ~ package, ~ It~ is~ already~ defined.
5513 }
5514 {
5515 The ~ anskey* ~ environment ~ is ~ defined ~ internally ~
5516 for ~ the ~ 'save-ans' ~ key.\\
5517 }

```

Message used in the creation of *prop list* by **enumext** package.

```

5518 \msg_new:nnn { enumext } { store-prop }
5519 {
5520 * ~ Package ~ enumext: ~ Creating ~
5521 \c_backslash_str g__enumext_#1_prop ~ \msg_line_context:.
5522 }
5523 \msg_new:nnn { enumext } { store-seq }
5524 {
5525 * ~ Package ~ enumext: ~ Creating ~
5526 \c_backslash_str g__enumext_#1_seq ~ \msg_line_context:.
5527 }
5528 \msg_new:nnn { enumext } { store-int }
5529 {
5530 * ~ Package ~ enumext: ~ Creating ~
5531 \c_backslash_str g__enumext_resume_#1_int ~ \msg_line_context:.
5532 }
5533 \msg_new:nnn { enumext } { prop-seq-int-hook }
5534 {
5535 * ~ Package ~ enumext: ~ Elements ~ in ~
5536 \c_backslash_str g__enumext_#1_prop ~ = ~ #2.\\
5537 * ~ Package ~ enumext: ~ Elements ~ in ~
5538 \c_backslash_str g__enumext_#1_seq ~ = ~ #3.\\
5539 * ~ Package ~ enumext: ~ Value ~ off ~
5540 \c_backslash_str g__enumext_resume_#1_int ~ = ~ #4.
5541 }
5542 \msg_new:nnn { enumext } { item-answer-hook }
5543 {
5544 * ~ Package ~ enumext: ~ Value ~ off ~
5545 \c_backslash_str g__enumext_item_number_int ~ = ~ #1.\\
5546 * ~ Package ~ enumext: ~ Value ~ off ~
5547 \c_backslash_str g__enumext_item_anskey_int ~ = ~ #2.\\
5548 * ~ Package ~ enumext: ~ Difference ~ item_number_int ~ - ~ item_anskey_int ~ = ~ #3.
5549 }

```

Message used by [*key* = *val*] system and `\setenumext` command.

```
5550 \msg_new:nnn { enumext } { invalid-key }
5551 {
5552 The ~ key ~ '#1' ~ is ~ not ~ know ~ the ~ level ~ #2.
5553 }
5554 \msg_new:nnn { enumext } { unknown-key-family }
5555 {
5556 Unknown~key~family~`\l_keys_key_str'~for~enumext.
5557 }
```

Messages used in length calculation.

```
5558 \msg_new:nnn { enumext } { width-negative }
5559 {
5560 Ignoring ~ negative ~ value ~ '#1=#2' ~ \msg_line_context:.\
5561 The ~ key ~ '#1'~ accepts ~ values ~ >= ~ 0pt.
5562 }
5563 \msg_new:nnn { enumext } { width-zero }
5564 {
5565 Invalid ~ '#1=#2' ~ \msg_line_context:.\
5566 The ~ key ~ '#1'~ accepts ~ values ~ > ~ 0pt.
5567 }
```

Messages used by `show-length` key in `enumext`.

```
5568 \msg_new:nnn { enumext } { list-lengths }
5569 {
5570 **** ~ Lengths ~ used ~ by ~ 'enumext' ~ level ~ '#2' ~ \msg_line_context:~\c_space_tl ****\
5571 __enumext_show_length:nnn { dim } { labelsep } {#1}
5572 __enumext_show_length:nnn { dim } { labelwidth } {#1}
5573 __enumext_show_length:nnn { dim } { itemindent } {#1}
5574 __enumext_show_length:nnn { dim } { leftmargin } {#1}
5575 __enumext_show_length:nnn { dim } { rightmargin } {#1}
5576 __enumext_show_length:nnn { dim } { listparindent } {#1}
5577 __enumext_show_length:nnn { skip } { topsep } {#1}
5578 __enumext_show_length:nnn { skip } { parsep } {#1}
5579 __enumext_show_length:nnn { skip } { partopsep } {#1}
5580 __enumext_show_length:nnn { skip } { itemsep } {#1}
5581 *****
5582 }
```

Messages used by `show-length` key in `enumext*`, `keyans*` and `keyans`.

```
5583 \msg_new:nnn { enumext } { list-lengths-not-nested }
5584 {
5585 **** ~ Lengths ~ used ~ by ~ '#2' ~ environment ~ \msg_line_context:~\c_space_tl ****\
5586 __enumext_show_length:nnn { dim } { labelsep } {#1}
5587 __enumext_show_length:nnn { dim } { labelwidth } {#1}
5588 __enumext_show_length:nnn { dim } { itemindent } {#1}
5589 __enumext_show_length:nnn { dim } { leftmargin } {#1}
5590 __enumext_show_length:nnn { dim } { rightmargin } {#1}
5591 __enumext_show_length:nnn { dim } { listparindent } {#1}
5592 __enumext_show_length:nnn { skip } { topsep } {#1}
5593 __enumext_show_length:nnn { skip } { parsep } {#1}
5594 __enumext_show_length:nnn { skip } { partopsep } {#1}
5595 __enumext_show_length:nnn { skip } { itemsep } {#1}
5596 *****
5597 }
```

Messages used by `ref` key.

```
5598 \msg_new:nnn { enumext } { key-ref-empty }
5599 {
5600 Key ~ 'ref' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
5601 }
```

Messages used by `save-ans` key.

```
5602 \msg_new:nnn { enumext } { save-ans-empty }
5603 {
5604 Key ~ 'save-ans' ~ need ~ a ~ value ~ in ~ '#1'~ \msg_line_context:.
5605 }
5606 \msg_new:nnn { enumext } { save-ans-log }
5607 {
5608 * ~ Package ~ enumext: ~ Start ~ #1\c_space_tl with ~ save-ans=#2 ~ \msg_line_context:.
5609 }
5610 \msg_new:nnn { enumext } { save-ans-log-hook }
5611 {
```

```

5612 * ~ Package ~ enumext: ~ Stop ~ #1\c_space_tl with ~ save-ans=#2 ~ \msg_line_context:.
5613 }
5614 \msg_new:nnn { enumext } { save-ans-hook }
5615 {
5616 Stop ~ storing ~ for ~ 'save-ans=#1' ~ \msg_line_context:.
5617 }

```

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

```

5618 \msg_new:nnn { enumext } { need-save-ans }
5619 {
5620 Key ~ '#1'~ works ~ only ~ with ~ the ~ 'save-ans' ~ key ~ in ~ '#2'~ \msg_line_context:.
5621 }
5622 \msg_new:nnn { enumext } { items-same-answer }
5623 {
5624 *****\
5625 * ~ Package ~ enumext: ~ Checking ~ answers ~ in ~ '#1' ~
5626 for ~ \c_left_brace_str #2 \c_right_brace_str\
5627 * ~ started ~ #3 ~ and ~ close ~ \msg_line_context: : ~
5628 'OK', ~ all ~ items ~ with ~ answer.\
5629 *****
5630 }
5631 \msg_new:nnn { enumext } { item-greater-answer }
5632 {
5633 Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\
5634 started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\
5635 Items ~ > ~ Answers.
5636 }
5637 \msg_new:nnn { enumext } { item-less-answer }
5638 {
5639 Checking ~ answers ~ in ~ '#1' ~ for ~ \c_left_brace_str #2 \c_right_brace_str\
5640 started ~ #3 ~ and ~ close ~ \msg_line_context: : ~'NOT ~ OK'\
5641 Items ~ < ~ Answers.
5642 }

```

Messages used by the internal system to check for “starred” `\item*` and `\anspic*` commands.

```

5643 \msg_new:nnn { enumext } { missing-starred }
5644 {
5645 Missing ~ '\c_backslash_str #1*' ~ #2.
5646 }
5647 \msg_new:nnn { enumext } { many-starred }
5648 {
5649 Many ~ '\c_backslash_str #1*' ~ #2.
5650 }

```

Messages used by `\printkeyans*` command.

```

5651 \msg_new:nnn { enumext } { print-starred }
5652 {
5653 \c_backslash_str printkeyans*:~ The ~ sequence ~ '#1' ~ already ~ contains ~
5654 #2 ~ environment ~ \msg_line_context:.
5655 }

```

Message for the nesting depth of the environment `enumext`.

```

5656 \msg_new:nnn { enumext } { list-too-deep }
5657 {
5658 Too ~ deep ~ nesting ~ for ~ 'enumext' ~ \msg_line_context:~ \
5659 The ~ maximum ~ level ~ of ~ nesting ~ is ~ 4.
5660 }

```

Messages used by `\anskey`, `anskey*` and `\anspic` commands.

```

5661 \msg_new:nnn { enumext } { anskey-unnumber-item }
5662 {
5663 Can't ~ store ~ with ~ a ~ unnumbered ~ \c_backslash_str item ~ \msg_line_context:.
5664 }
5665 \msg_new:nnn { enumext } { anskey-already-stored }
5666 {
5667 Content ~ already ~ stored ~ for ~ this ~ \c_backslash_str item ~ \msg_line_context:.
5668 }
5669 \msg_new:nnn { enumext } { anskey-empty-arg }
5670 {
5671 Can't ~ store ~ empty ~ content ~ \msg_line_context:.
5672 }
5673 \msg_new:nnn { enumext } { anskey-wrong-place }
5674 {

```

```

5675 Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:~ \\
5676 '\c_backslash_str #1' ~ works ~ in ~ the ~ environment ~ '#2'.
5677 }
5678 \msg_new:nnn { enumext } { anskey-nested }
5679 {
5680 The ~ command ~ \c_backslash_str anskey~ can't ~ be ~ nested ~ \msg_line_context:.
5681 }
5682 \msg_new:nnn { enumext } { anskey-math-mode }
5683 {
5684 #1 ~ can't ~ work ~ in ~ math ~ mode ~ \msg_line_context:.
5685 }
5686 \msg_new:nnn { enumext } { anskey-env-wrong }
5687 {
5688 The ~ environment ~ anskey* ~ cannot ~ use ~ in ~ '#1' ~ \msg_line_context:.
5689 }
5690 \msg_new:nnn { enumext } { ans-pic-wrong-place }
5691 {
5692 Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:~ \\
5693 '\c_backslash_str #1' ~ works ~ in ~ the ~ environment ~ '#2'.
5694 }
5695 \msg_new:nnn { enumext } { command-wrong-place }
5696 {
5697 Wrong ~ place ~ for ~ command ~ '\c_backslash_str #1' ~ \msg_line_context:~ \\
5698 '\c_backslash_str #1' ~ works ~ outside ~ the ~ environment ~ '#2'.
5699 }
5700 \msg_new:nnnn { enumext } { anskey-env-key-unknown }
5701 {
5702 The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment~
5703 'anskey*' ~ and ~ is ~ being ~ ignored.
5704 }
5705 {
5706 The ~ environment ~ 'anskey*' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5707 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5708 }
5709 \msg_new:nnnn { enumext } { anskey-env-key-value-unknown }
5710 {
5711 The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~
5712 'anskey*' ~ and ~ is ~ being ~ ignored.
5713 }
5714 {
5715 The ~ environment ~ 'anskey*' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5716 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5717 }
5718 \msg_new:nnnn { enumext } { anskey-cmd-key-unknown }
5719 { The ~ key ~ '#1' ~ is ~ unknown ~ by ~ '\c_backslash_str anskey' ~ and ~ is ~ being ~ ignored.}
5720 {
5721 The ~ command ~ '\c_backslash_str anskey' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5722 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5723 }
5724 \msg_new:nnnn { enumext } { anskey-cmd-key-value-unknown }
5725 { The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ '\c_backslash_str anskey' ~ and ~ is ~ being ~ ignored.}
5726 {
5727 The ~ command ~ '\c_backslash_str anskey' ~ does ~ not ~ have ~ a ~ key ~ called ~'#1'.\\
5728 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5729 }

```

Messages used by **keyans**, **keyans\*** and **keyanspic** environment.

```

5730 \msg_new:nnn { enumext } { keyans-nested }
5731 {
5732 The ~ environment ~ 'keyans' ~ can't ~ be ~ nested ~ \msg_line_context:.
5733 }
5734 \msg_new:nnn { enumext } { keyans-wrong-level }
5735 {
5736 Wrong ~ level ~ position ~ for ~ 'keyans' ~ \msg_line_context:~ \\
5737 The ~ environment ~ 'keyans' ~ can ~ only ~ be ~ in ~ the ~ first ~ level.
5738 }
5739 \msg_new:nnn { enumext } { wrong-place }
5740 {
5741 Wrong ~ place ~ for ~ '#1' ~ environment ~ \msg_line_context:~ \\
5742 '#1' ~ is ~ only ~ found ~ with ~ '#2' ~ in ~ 'enumext.
5743 }
5744 \msg_new:nnn { enumext } { keyanspic-nested }

```

```

5745 {
5746 The ~ environment ~ 'keyanspic' ~ can't ~ be ~ nested~ \msg_line_context:~.
5747 }
5748 \msg_new:nnn { enumext } { keyanspic-wrong-level }
5749 {
5750 Wrong ~ level ~ position ~ for ~ 'keyanspic' ~ \msg_line_context:~ \\
5751 The ~ environment ~ 'keyans' ~ can ~ only ~ be ~ in ~ the ~ first ~ level.
5752 }
5753 \msg_new:nnn { enumext } { keyanspic-item-cmd }
5754 {
5755 Can't ~ use ~ \c_backslash_str item ~ in ~ keyanspic ~ \msg_line_context:.
5756 }
5757 \msg_new:nnnn { enumext } { keyans-unknown-key }
5758 {
5759 The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment~
5760 '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5761 }
5762 {
5763 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5764 ~ have ~ a ~ key ~ called ~'#1'.\\
5765 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5766 }
5767 \msg_new:nnnn { enumext } { keyans-unknown-key-value }
5768 {
5769 The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~
5770 '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5771 }
5772 {
5773 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5774 ~ have ~ a ~ key ~ called ~'#1'.\\
5775 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5776 }

```

Message used by unknown *⟨keys⟩* in *enumext\**. environment.

```

5777 \msg_new:nnnn { enumext } { starred-unknown-key }
5778 {
5779 The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment~
5780 '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5781 }
5782 {
5783 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5784 ~ have ~ a ~ key ~ called ~'#1'.\\
5785 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5786 }
5787 \msg_new:nnnn { enumext } { starred-unknown-key-value }
5788 {
5789 The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~
5790 '\l__enumext_envir_name_tl' ~ and ~ is ~ being ~ ignored.
5791 }
5792 {
5793 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5794 ~ have ~ a ~ key ~ called ~'#1'.\\
5795 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5796 }

```

Message used by unknown *⟨keys⟩* in *enumext* environment.

```

5797 \msg_new:nnnn { enumext } { standar-unknown-key }
5798 {
5799 The ~ key ~ '#1' ~ is ~ unknown ~ by ~ environment ~ '\l__enumext_envir_name_tl' \c_space_tl
5800 ~ on ~ level ~ \int_use:N \l__enumext_level_int \c_space_tl and ~ is ~ being ~ ignored.
5801 }
5802 {
5803 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5804 ~ have ~ a ~ key ~ called ~'#1' ~ on ~ level ~ \int_use:N \l__enumext_level_int.\\
5805 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5806 }
5807 \msg_new:nnnn { enumext } { standar-unknown-key-value }
5808 {
5809 The ~ key ~ '#1=#2' ~ is ~ unknown ~ by ~ environment ~ '\l__enumext_envir_name_tl' \c_space_
5810 ~ on ~ level ~ \int_use:N \l__enumext_level_int \c_space_tl and ~ is ~ being ~ ignored.
5811 }
5812 {

```

```

5813 The ~ environment ~ '\l__enumext_envir_name_tl' ~ does ~ not
5814 ~ have ~ a ~ key ~ called ~ '#1' ~ on ~ level ~ \int_use:N \l__enumext_level_int.\\
5815 Check ~ that ~ you ~ have ~ spelled ~ the ~ key ~ name ~ correctly.
5816 }

```

Message used by unknown *⟨keys⟩* in `\foreachkeyans`.

```

5817 \msg_new:nnnn { enumext } { for-key-unknown }
5818 { The~key~'#1'~is~unknown~by~'\c_backslash_str foreachkeyans'~and~is~being~ignored.}
5819 {
5820 The~command~'\c_backslash_str foreachkeyans'~does~not~have~a~key~called~'#1'.\\
5821 Check~that~you~have~spelled~the~key~name~correctly.
5822 }
5823 \msg_new:nnnn { enumext } { for-key-value-unknown }
5824 { The~key~'#1=#2'~is~unknown~by~'\c_backslash_str foreachkeyans'~and~is~being~ignored. }
5825 {
5826 The~command~'\c_backslash_str foreachkeyans'~does~not~have~a~key~called~'#1'.\\
5827 Check~that~you~have~spelled~the~key~name~correctly.
5828 }

```

Messages used by `\getkeyans` command.

```

5829 \msg_new:nnn { enumext } { undefined-storage-anskey }
5830 {
5831 Storage ~ named ~ '#1' ~ is ~ not ~ defined ~ \msg_line_context:.
5832 }

```

Messages used by `\miniright` command.

```

5833 \msg_new:nnn { enumext } { missing-miniright }
5834 {
5835 Missing ~ '\c_backslash_str miniright' ~ in ~ \msg_line_context:.\\
5836 The ~ key ~ 'mini-env' ~ need ~ '\c_backslash_str miniright'.
5837 }
5838 \msg_new:nnn { enumext } { wrong-miniright-place }
5839 {
5840 Wrong ~ place ~ for ~ '\c_backslash_str miniright' ~ \msg_line_context:.~ \\
5841 Works ~ in ~ 'enumext' ~ and ~ 'keyans' ~ with ~ key ~ 'mini-env'.
5842 }
5843 \msg_new:nnn { enumext } { wrong-miniright-use }
5844 {
5845 Wrong ~ use ~ for ~ '\c_backslash_str miniright' ~ \msg_line_context:.~ \\
5846 '\c_backslash_str miniright' ~ need ~ a ~ key ~ 'mini-env'.
5847 }
5848 \msg_new:nnn { enumext } { wrong-miniright-starred }
5849 {
5850 Can't ~ use ~ \c_backslash_str miniright ~ in ~ starred ~ environments ~ \msg_line_context:.
5851 }
5852 \msg_new:nnn { enumext } { many-miniright-used }
5853 {
5854 Can't ~ use ~ \c_backslash_str miniright ~ more ~ than ~ once ~ \msg_line_context:.
5855 }

```

Messages used by `\setenumextmeta` command.

```

5856 \msg_new:nnn { enumext } { unknown-set }
5857 {
5858 Argument ~ [#1] ~ is ~ unknown ~ by ~ \c_backslash_str setenumextmeta ~ \msg_line_context:.
5859 }
5860 \msg_new:nnn { enumext } { already-defined }
5861 {
5862 The ~ key ~ '#1' ~ is ~ already ~ defined ~ \msg_line_context:.
5863 }
5864 \msg_new:nnn { enumext } { prohibited-unknown }
5865 {
5866 The ~ name ~ 'unknown' ~ can't ~ be ~ chosen~ for ~ a ~ meta ~ key ~ \msg_line_context:.
5867 }

```

Messages used by `enumext*` and `keyans*` environments.

```

5868 \msg_new:nnn { enumext } { nested }
5869 {
5870 The ~ environment ~ \l__enumext_envir_name_tl \c_space_tl can't ~ be ~ nested ~ \msg_line_con
5871 }
5872 \msg_new:nnn { enumext } { nested-horizontal }
5873 {
5874 The ~ environment ~ \l__enumext_envir_name_tl \c_space_tl can't ~ be ~ nested ~ in ~ '#1' ~
5875 }

```



```
5876 \msg_new:nnn { enumext } { item-joined }
5877 {
5878 Items ~ joined ~ (#1) ~ > ~ #2 ~ columns ~\msg_line_context:.
5879 }
5880 \msg_new:nnn { enumext } { item-joined-columns }
5881 {
5882 Not ~ space ~ to ~ join ~ items ~ (#1) ~ > ~ #2 ~\msg_line_context:.
5883 }
```

## 13.52 Finish package

Finish package implementation.

```
5884 \file_input_stop:
5885 </package>
```

# 14 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                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                 |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| <code>\*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 229                             |
| <code>\+</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 221                             |
| <code>\-</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 221                             |
| <code>\\</code> 237, 2904, 4295, 4298, 5502, 5511, 5516, 5536, 5538, 5545, 5547, 5560, 5565, 5570, 5585, 5624, 5626, 5628, 5633, 5634, 5639, 5640, 5658, 5675, 5692, 5697, 5706, 5715, 5721, 5727, 5736, 5741, 5750, 5764, 5774, 5784, 5794, 5804, 5814, 5820, 5826, 5835, 5840, 5845                                                                                                                                                                                                                                                                                                                                                                 |                                 |
| A                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>above</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | <u>1716</u>                     |
| <code>above*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | <u>1716</u>                     |
| <code>\addvspace</code> 1283, 1311, 1354, 1357, 1525, 1528, 1625, 1631, 1669, 1675, 1696, 1702, 3747, 3908, 3926, 4186, 4190, 4543, 4558, 4604, 4618                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                 |
| <code>after</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | <u>1113</u>                     |
| <code>align</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | <u>666</u>                      |
| <code>\Alph</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 41, 45, <u>46</u>               |
| <code>\Alpha</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 608, 736, 780, 846, 5210        |
| <code>\alph</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 41, 45, <u>46</u>               |
| <code>\alpha</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 609, 734, 5202                  |
| <code>\anskey</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 13, 79, 81, <u>2722</u>         |
| <code>anskey*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 13, <u>2832</u>                 |
| <code>\anspic</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 16, 108, 111, <u>4200</u>       |
| <code>\anspic*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 73                              |
| <code>\arabic</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 33, <u>41</u>                   |
| <code>\arabic</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 607, 733, 779, 5194, 5198, 5214 |
| B                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>base-fix</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | <u>975</u>                      |
| <code>\baselineskip</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <u>54</u>                       |
| <code>\baselineskip</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 991, 998                        |
| <code>before</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | <u>1113</u>                     |
| <code>before*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | <u>1113</u>                     |
| <code>below</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | <u>1716</u>                     |
| <code>below*</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | <u>1716</u>                     |
| bool commands:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                 |
| <code>\bool_gset_false:N</code> 350, 351, 352, 3008, 3010, 4560, 4564, 4620                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                 |
| <code>\bool_gset_true:N</code> 258, 268, 1216, 2209, 2215, 4529, 4561, 4593, 4621                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>\bool_if:NTF</code> . 401, 411, 428, 502, 509, 518, 525, 539, 552, 1738, 1752, 1765, 1776, 1787, 1798, 1809, 1820, 1869, 1886, 1891, 1899, 1926, 1964, 1969, 1976, 1980, 2002, 2007, 2015, 2022, 2053, 2061, 2154, 2354, 2364, 2443, 2467, 2474, 2498, 2596, 2618, 2658, 2672, 2676, 2726, 2745, 2769, 2823, 2834, 2923, 2960, 3024, 3059, 3074, 3149, 3160, 3164, 3183, 3196, 3238, 3272, 3308, 3323, 3344, 3483, 3504, 3588, 3598, 3631, 3636, 3702, 3728, 3778, 3836, 3891, 3916, 4120, 4184, 4202, 4221, 4266, 4293, 4522, 4538, 4544, 4587, 4601, 4605, 4694, 4704, 4792, 4798, 4805, 4821, 4914, 4924, 5037, 5041, 5069, 5085, 5093, 5121 |                                 |
| <code>\bool_if:nTF</code> 1676, 1703, 3294, 3462, 3546, 4242, 5233, 5377                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                 |
| <code>\bool_if_p:N</code> 277, 292, 985, 986, 994, 995, 1648, 2033, 2034, 2042, 2043, 2167, 2193, 2206, 2207, 2212, 2213, 2531, 2541, 2553, 2568, 2569, 2603, 2644, 2645, 2946, 3136, 3137, 3174, 3175, 3496, 3497, 3675, 3677, 3688, 4249, 4250, 5061, 5062                                                                                                                                                                                                                                                                                                                                                                                          |                                 |
| <code>\bool_lazy_all:nTF</code> 275, 290, 983, 2165, 2191, 2529, 2538, 2551, 2566, 3494, 3673, 3686, 5059                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                 |
| <code>\bool_lazy_and:nnTF</code> 254, 264, 993, 1640, 1647, 2032, 2041, 2205, 2211, 2602, 2609, 2643, 2787, 2799, 2945, 2951, 3135                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |                                 |
| <code>\bool_lazy_or:nnTF</code> . . 2094, 2101, 3173, 4248, 5400                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                 |
| <code>\bool_new:N</code> 30, 31, 32, 33, 34, 35, 36, 37, 60, 69, 93, 98, 99, 104, 105, 108, 127, 134, 135, 142, 149, 150, 153, 157, 159, 160, 177, 189, 191                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                 |
| <code>\bool_not_p:n</code> 255, 265, 987, 1649, 2540, 2604, 2610, 2947, 2952, 3676, 3689                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                 |
| <code>\bool_set_eq:NN</code> . . . . . 3247, 3441, 4745, 4989                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                 |
| <code>\bool_set_false:N</code> 408, 1005, 2139, 2140, 2172, 2177, 2181, 2185, 2198, 2887, 3650, 3795, 3844, 3931, 4073, 4117, 4663, 4689, 4742, 4930, 4985, 5246, 5247                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                 |
| <code>\bool_set_true:N</code> . 282, 283, 297, 298, 393, 396, 659, 1020, 1722, 1727, 1989, 2111, 2112, 2386, 2394, 2888, 3241, 3243, 3275, 3277, 3437, 3449, 3611, 3649, 3682, 3695, 3768, 3841, 3868, 4070, 4511, 4576, 4662, 4749, 4756, 4757, 4801, 4928, 4993, 5000, 5002, 5241, 5242                                                                                                                                                                                                                                                                                                                                                             |                                 |
| box commands:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                 |
| <code>\box_dp:N</code> . . 1571, 1572, 1575, 1582, 1595, 1603, 1609, 1617, 4131, 4136, 4186, 4277                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>\box_ht:N</code> . . 1354, 1357, 1368, 1369, 1380, 1382, 1397, 1400, 1408, 1409, 1420, 1422, 1437, 1440, 1447, 1448, 1459, 1461, 1476, 1479, 1525, 1528, 1536, 1537, 1545, 1546, 1558, 1560                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>\box_ht_plus_dp:N</code> . . . . . 4126, 4194, 4230                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                 |
| <code>\box_new:N</code> . . . . . 66, 145, 146, 184, 190                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                 |
| <code>\box_use_drop:N</code> . . . . . 4555, 4616, 4857, 5133                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                 |
| <code>\box_wd:N</code> . . . . . 615                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                 |
| C                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                 |
| <code>\c</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 229, 230, 882, 884, 896, 898    |
| <code>\catcode</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 2904                            |
| <code>\cB</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 230                             |
| <code>\cE</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 230                             |
| <code>\centering</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | 1678, 1705, 4321, 4548, 4609    |
| <code>check-ans</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | <u>2131</u>                     |
| Document class:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                 |
| <code>article</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 47                              |
| clist commands:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                 |
| <code>\clist_const:Nn</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 196                             |
| <code>\clist_map_function:nN</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | 4304                            |
| <code>\clist_map_inline:Nn</code> . 665, 930, 1112, 1127, 1208, 1732                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                 |
| <code>\clist_map_inline:nn</code> . 45, 56, 74, 82, 95, 107, 137, 168, 195, 643, 696, 716, 1025, 1046, 1222, 1838, 2078, 2145, 2333, 2351, 2383, 2526, 3068, 3366, 3378, 3418, 3575, 3578, 3606, 3618, 3621, 3641, 5353                                                                                                                                                                                                                                                                                                                                                                                                                               |                                 |
| <code>\columnbreak</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 79                              |
| <code>\columnbreak</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 2606                            |
| <code>columns</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | <u>1192</u>                     |
| <code>columns-sep</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | <u>1192</u>                     |
| <code>\columnsep</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | 102                             |
| <code>\columnsep</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | 3723, 3889                      |
| <code>\columnseprule</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 102                             |
| <code>\columnseprule</code> .....                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 3726, 3890                      |

Commands provide by **enumext**:

\anskey . 30, 69, 70, 75, 76, 78–82, 88, 90, 101, 120, 129, 131, 138  
 \anspic\* . . . . . 30, 31, 73, 76, 88, 89, 111, 129, 131  
 \anspic . . . . . 30, 76, 108, 111, 138  
 \foreachkeyans . . . . . 134, 141  
 \getkeyans . . . . . 76, 129, 141  
 \item\* 30, 31, 73, 76, 88, 89, 92, 96, 121, 122, 127, 129, 131  
 \item . . . . . 92, 96, 115, 121, 123, 126  
 \miniright . . . . . 29, 52, 60, 61, 103, 141  
 \printkeyans\* . . . . . 130  
 \printkeyans . . . . . 30, 76, 130, 131  
 \setenumextmeta . . . . . 133, 141  
 \setenumext . . . . . 30, 131–133, 137

Counters defined by **enumext**:

enumXiii . . . . . 28, 40  
 enumXii . . . . . 28, 40  
 enumXiv . . . . . 28, 40  
 enumXi . . . . . 28, 40  
 enumXviii . . . . . 28, 40  
 enumXvii . . . . . 28, 40, 122  
 enumXvi . . . . . 28, 40  
 enumXv . . . . . 28, 40

cs commands:

\cs\_generate\_variant:Nn . 201, 202, 617, 633, 888, 904, 2435, 2440, 2516, 2840, 3565, 4306, 5412  
 \cs\_if\_exist:NTF . . . . . 587  
 \cs\_if\_exist\_p:N . . . . . 3499, 5064  
 \cs\_if\_free:NTF . . . . . 2791, 2803  
 \cs\_new:Nn . . . . . 215  
 \cs\_new:Npn . 233, 1839, 1848, 1856, 2398, 2407, 2415, 5261, 5270, 5279  
 \cs\_new\_eq:NN . 377, 378, 383, 384, 413, 414, 417, 418  
 \cs\_new\_protected:Nn . 225, 239, 247, 273, 306, 336, 342, 348, 354, 360, 368, 388, 436, 440, 458, 470, 488, 500, 516, 532, 545, 566, 756, 817, 868, 981, 1128, 1132, 1136, 1140, 1144, 1148, 1152, 1156, 1160, 1164, 1168, 1172, 1176, 1180, 1184, 1188, 1223, 1235, 1268, 1285, 1296, 1313, 1339, 1360, 1485, 1511, 1531, 1564, 1586, 1621, 1627, 1733, 1747, 1761, 1772, 1783, 1794, 1805, 1816, 1897, 2000, 2013, 2030, 2051, 2079, 2084, 2109, 2150, 2160, 2203, 2218, 2225, 2234, 2239, 2244, 2249, 2258, 2263, 2268, 2441, 2465, 2472, 2496, 2503, 2517, 2743, 2762, 2778, 2841, 2877, 2908, 2943, 2985, 3006, 3014, 3057, 3072, 3100, 3133, 3169, 3181, 3194, 3280, 3290, 3301, 3317, 3333, 3458, 3476, 3510, 3520, 3642, 3671, 3700, 3707, 3737, 3754, 3776, 3798, 3834, 3858, 3875, 3900, 3914, 3935, 4092, 4288, 4302, 4307, 4331, 4341, 4372, 4501, 4520, 4566, 4585, 4649, 4676, 4683, 4692, 4702, 4727, 4868, 4912, 4943, 4949, 4970, 5029, 5144  
 \cs\_new\_protected:Npn 203, 207, 211, 421, 585, 602, 612, 618, 737, 781, 851, 875, 889, 1660, 1689, 1865, 1884, 1954, 1987, 2089, 2273, 2352, 2362, 2384, 2392, 2427, 2436, 2592, 2655, 2670, 2708, 2712, 2832, 2863, 2867, 2898, 3034, 3110, 3154, 3234, 3253, 3379, 3383, 3397, 3401, 3419, 3423, 3433, 3445, 3492, 3534, 3568, 3609, 3653, 3854, 4101, 4108, 4115, 4219, 4238, 4262, 4403, 4452, 4666, 4733, 4740, 4754, 4762, 4767, 4777, 4936, 4976, 4983, 4998, 5007, 5024, 5057, 5166, 5179, 5227, 5350, 5362, 5386, 5398, 5436, 5446, 5454, 5476  
 \cs\_new\_protected\_nopar:Nn . . 3998, 4042, 4050, 4058, 4712, 4720, 4851, 4955, 4963, 5127  
 \cs\_new\_protected\_nopar:Npn . . 3990, 4006, 4783, 4829, 5075, 5101

\cs\_set:Npn . . . . . 2527, 2564, 5172  
 \cs\_set\_eq:NN . . 3502, 4639, 4640, 4831, 4901, 4902, 5067, 5103  
 \cs\_set\_protected:Nn . . . . 1051, 1067, 1080, 1092  
 \cs\_set\_protected:Npn . 41, 50, 67, 75, 90, 96, 130, 164, 175, 634, 644, 666, 701, 717, 763, 905, 931, 1007, 1030, 1104, 1113, 1192, 1209, 1716, 1827, 2070, 2131, 2290, 2334, 2370, 2519, 3061, 3355, 3371, 3411, 3566, 3607  
 \cs\_to\_str:N . . . . . 604, 627  
 \cs\_undefine:N . . . . . 2780, 2781, 2782, 2783

## D

\d . . . . . 221  
 \DeclareDocumentEnvironment . . . . . 570  
 dim commands:  
 \dim\_abs:n . . . . . 3539, 3544  
 \dim\_add:Nn . . . . . 4135, 4366, 4397  
 \dim\_compare:nNnTF . . 1053, 1069, 1082, 1094, 1372, 1384, 1412, 1424, 1451, 1463, 1540, 1548, 1662, 1691, 3536, 3541, 3547, 3553, 3555, 3557, 3712, 3759, 3862, 3879, 4110, 4343, 4359, 4374, 4390, 4503, 4568  
 \dim\_compare:nTF . . . . . 2628, 2973, 3801, 3938  
 \dim\_eval:n . . . . . 991, 4192, 4273  
 \dim\_gset\_eq:NN . . . . . 4512, 4577  
 \dim\_gzero:N . . . . . 3012, 4563, 4623  
 \dim\_new:N . 63, 70, 71, 72, 92, 139, 147, 148, 183, 185, 186, 192  
 \dim\_set:Nn . 615, 1021, 3270, 3539, 3544, 3546, 3549, 3550, 3554, 3556, 3559, 3560, 3562, 3715, 3762, 3800, 3864, 3881, 3937, 4124, 4228, 4309, 4345, 4352, 4376, 4383, 4438, 4487, 4505, 4570, 4779  
 \dim\_set\_eq:NN 724, 770, 839, 843, 3185, 3186, 3198, 3199, 3265, 3577, 3620, 3723, 3889, 4445, 4448, 4449, 4494, 4497, 4498, 4772, 4843, 5115  
 \dim\_sub:Nn . . . . . 3806, 3943, 4361, 4392  
 \dim\_use:N . 1054, 1062, 1663, 1673, 2506, 2509, 2514, 3285, 3287, 3340, 3713, 3717, 3718, 3720, 3760, 3765, 3766, 3772, 3803, 3808  
 \dim\_zero:N . . . . . 3612, 3726, 3890, 4137  
 \dim\_zero\_new:N . . . . . 584  
 \c\_zero\_dim 1056, 1070, 1083, 1095, 1663, 1691, 2630, 2975, 3536, 3541, 3547, 3554, 3713, 3760, 3803, 3862, 3879, 3940, 4110, 4343, 4359, 4374, 4390, 4503, 4568  
 \dimeval . . . . . 2297

## E

\end . . . 2469, 2500, 3744, 3905, 4174, 4323, 5235, 5245, 5253  
 end internal commands:  
 \end\_\_enumext\_\_mini\_page . 1671, 1698, 3787, 3925, 4527, 4591, 4617  
 \endgroup . . . . . 2904  
 \endlist . . . . . 378  
 \endminipage . . . . . 384  
 enumext . . . . . 5, 3812  
 enumext internal commands:  
 \l\_\_enumext\_\_ref\_the\_count\_tl . . . . . 43  
 \l\_\_enumext\_\_resume\_name\_tl . . . . . 65  
 \\_\_enumext\_add\_meta\_key:nnn . . 134, 5364, 5380, 5381, 5383, 5386  
 \\_\_enumext\_add\_pre\_parsep: . 53, 1233, 1235, 1235  
 \\_\_enumext\_after\_args\_exec: 51, 1128, 1140, 3825  
 \\_\_enumext\_after\_args\_exec\_v: 1144, 1156, 3958  
 \\_\_enumext\_after\_args\_exec\_viii: . . 1160, 1184  
 \\_\_enumext\_after\_args\_exec\_viii: . . . . . 1188

`\__enumext_after_env:nn` 85, 86, 88, 104, 117, 124, 207, 207, 558, 562, 2918, 3830, 4536, 4599, 4884  
`\__enumext_after_hyperref:` ... 36, 386, 386, 388  
`\l__enumext_after_list_args_v_tl` ..... 1158  
`\l__enumext_after_list_args_vii_tl` 1186, 4849  
`\l__enumext_after_list_args_viii_tl` .. 1190, 5125  
`\__enumext_after_list_vii:` 117, 120, 4647, 4683, 4683  
`\__enumext_after_list_viii:` ... 125, 4910, 4949, 4949  
`\__enumext_after_stop_list:` 51, 103, 1128, 1136, 3792  
`\__enumext_after_stop_list_v:` 1144, 1152, 3932  
`\l__enumext_after_stop_list_v_tl` ..... 1154  
`\__enumext_after_stop_list_vii:` .. 120, 1160, 1176, 4686  
`\l__enumext_after_stop_list_vii_tl` ... 1178  
`\__enumext_after_stop_list_viii:` . 1180, 4952  
`\l__enumext_after_stop_list_viii_tl` ... 1182  
`\l__enumext_align_label_pos_v_str` .... 3526  
`\l__enumext_align_label_pos_X_str` ..... 75  
`\l__enumext_align_label_vii_str` ..... 4818  
`\l__enumext_align_label_viii_str` ..... 5089  
`\l__enumext_align_label_X_str` ..... 175  
`\c__enumext_all_envs_clist` . 196, 665, 930, 1112, 1127, 1208, 1732  
`\c__enumext_all_families_seq` .. 133, 5318, 5344  
`\l__enumext_anskey_env_bool` 34, 84, 30, 283, 298, 2834  
`\__enumext_anskey_env_clean_vars:` . 87, 2939, 2943, 3006  
`\__enumext_anskey_env_define_keys:` 84, 2832, 2841, 2912  
`\__enumext_anskey_env_exec:` 85, 2837, 2908, 2908  
`\__enumext_anskey_env_make:n` 69, 84, 2114, 2832, 2832, 2840  
`\__enumext_anskey_env_reset_keys:` 85, 86, 2832, 2877, 2940  
`\__enumext_anskey_env_save_keys:` .. 86, 2920, 2943, 2943  
`\__enumext_anskey_env_store:` .. 87, 2936, 2943, 2985  
`\__enumext_anskey_env_unknown:n` 85, 2832, 2860, 2863  
`\__enumext_anskey_env_unknown:nn` . 2832, 2865, 2867  
`\l__enumext_anskey_level_int` .. 24, 2764, 2765  
`\__enumext_anskey_safe_inner:` . 83, 2737, 2743, 2762  
`\__enumext_anskey_safe_inner:n` ..... 82  
`\__enumext_anskey_safe_outer:` . 82, 2724, 2743, 2743  
`\__enumext_anskey_show_wrap_arg:n` . 80, 2655, 2655, 2674, 2689  
`\__enumext_anskey_show_wrap_left:n` 81, 2600, 2670, 2670  
`\__enumext_anskey_unknown:n` 81, 2692, 2706, 2708  
`\__enumext_anskey_unknown:nn` . 2692, 2710, 2712  
`\__enumext_anskey_wrapper:n` ..... 2294, 2668  
`\l__enumext_anspic_above_int` . 138, 4310, 4311, 4313  
`\__enumext_anspic_args:nnn` 111, 113, 4200, 4216, 4288  
`\l__enumext_anspic_args_seq` 111, 113, 138, 4214, 4322, 4335  
`\l__enumext_anspic_below_int` . 138, 4310, 4311, 4314  
`\l__enumext_anspic_body_box` ... 138, 4227, 4230  
`\__enumext_anspic_body_dim:n` .. 111, 4200, 4219, 4265  
`\l__enumext_anspic_body_htdp_dim` .. 111, 138, 4228, 4276  
`\__enumext_anspic_exec:` ..... 4200  
`\__enumext_anspic_exec:` ... 110, 113, 4169, 4331  
`\__enumext_anspic_label:nn` 112, 4200, 4238, 4268, 4283  
`\l__enumext_anspic_label_above_bool` ... 138, 4070, 4073, 4120, 4184, 4221, 4266, 4293  
`\l__enumext_anspic_label_box` .. 138, 4123, 4126  
`\l__enumext_anspic_label_htdp_dim` . 110, 138, 4124, 4130, 4194, 4275  
`\__enumext_anspic_label_pos:nnn` .. 112, 4200, 4262, 4291  
`\l__enumext_anspic_label_sep_skip` 4080, 4132, 4195, 4278, 4295  
`\l__enumext_anspic_layout_style_tl` 4082, 4333, 4338  
`\l__enumext_anspic_mini_pos_str` .. 138, 4071, 4074, 4320  
`\l__enumext_anspic_mini_width_dim` 138, 4240, 4309, 4320  
`\__enumext_anspic_print:n` 113, 4200, 4302, 4306, 4335, 4338  
`\__enumext_anspic_row:n` .. 113, 4200, 4304, 4307  
`\__enumext_anspic_start_list_tag:` 4014, 4042, 4290  
`\__enumext_anspic_stop_list_tag:` . 4014, 4058, 4300  
`\__enumext_anspic_stop_start_list_tag:` 4014, 4050, 4292  
`\__enumext_at_begin_document:n` .. 36, 203, 203, 375, 381  
`\l__enumext_base_line_fix_bool` 48, 131, 977, 986, 1005, 5241, 5246  
`\__enumext_before_args_exec:` 50, 102, 120, 1128, 1128, 3757  
`\__enumext_before_args_exec_v:` 1144, 1144, 3861  
`\__enumext_before_args_exec_vii:` . 1160, 1160, 4680  
`\__enumext_before_args_exec_viii:` 1164, 4946  
`\__enumext_before_env:nn` 84, 207, 211, 2785, 2797, 2809, 2910  
`\__enumext_before_keys_exec:` .. 50, 1128, 1132, 3822  
`\__enumext_before_keys_exec_v:` 1144, 1148, 3955  
`\__enumext_before_keys_exec_vii` ..... 1160  
`\__enumext_before_keys_exec_vii:` . 1168, 4634  
`\__enumext_before_keys_exec_viii:` 1172, 4896  
`\__enumext_before_list:` .. 102, 3754, 3754, 3816  
`\__enumext_before_list_v:` ... 3858, 3858, 3950  
`\__enumext_before_list_vii:` ... 120, 4629, 4676, 4676  
`\__enumext_before_list_viii:` .. 125, 4892, 4943, 4943  
`\l__enumext_before_no_starred_key_v_tl` 1150  
`\l__enumext_before_no_starred_key_vii_-tl` ..... 1170

```

\l__enumext_before_no_starred_key_viii_-
 tl 1174
\l__enumext_before_starred_key_v_tl ... 1146
\l__enumext_before_starred_key_vii_tl . 1162
\l__enumext_before_starred_key_viii_tl 1166
__enumext_calc_hspace:NNNNNN 98, 3534, 3534,
 3565, 3570, 3613
__enumext_check_ans_active: 71, 102, 120, 2150,
 2150, 3758, 4679
\g__enumext_check_ans_item_tl 90
\g__enumext_check_ans_key_bool .. 72, 149, 350,
 2209, 2215, 3024
\l__enumext_check_ans_key_bool 72, 2135, 2140,
 2206, 2212
__enumext_check_ans_key_hook: .. 72, 103, 120,
 2203, 2203, 3793, 4687
__enumext_check_ans_level: 71, 2150, 2156, 2160
__enumext_check_ans_log: 72, 87, 2249, 2249, 3028
__enumext_check_ans_log_msg_greater: 2249,
 2255, 2268
__enumext_check_ans_log_msg_less: 2249, 2253,
 2258
__enumext_check_ans_log_msg_same_ok: 2249,
 2254, 2263
__enumext_check_ans_msg_greater: 2225, 2231,
 2244
__enumext_check_ans_msg_less: 2225, 2229, 2234
__enumext_check_ans_msg_same_ok: 2225, 2230,
 2239
__enumext_check_ans_show: .. 72, 87, 2225, 2225,
 3026
\l__enumext_check_answers_bool . 69, 71, 82, 92,
 149, 2112, 2139, 2154, 2443, 2467, 2474, 2498, 2726,
 2923, 3149, 3238, 3272, 4798
__enumext_check_starred_cmd:n 34, 73, 90, 124,
 2273, 2273, 3961, 4182, 4909
\g__enumext_check_starred_cmd_int .. 96, 149,
 2276, 2282, 2287, 3456, 4247, 5036
\l__enumext_check_start_line_env_tl . 34, 149,
 313, 321, 329, 2279, 2285, 2288
\l__enumext_columns_sep_v_dim 3879, 3881, 3889
\l__enumext_columns_sep_vii_dim .. 4343, 4345,
 4354, 4366, 4442, 4865
\l__enumext_columns_sep_viii_dim . 4374, 4376,
 4385, 4397, 4491, 5141
\l__enumext_columns_v_int 1505, 1523, 1694, 3877,
 3885, 3897, 3902
\l__enumext_columns_vii_int .. 4348, 4351, 4355,
 4364, 4406, 4410, 4413, 4419, 4425, 4429, 4859, 4873
\l__enumext_columns_viii_int . 4379, 4382, 4386,
 4395, 4455, 4459, 4462, 4468, 4474, 4478, 5135, 5150
\l__enumext_counter_i_tl 41, 594
\l__enumext_counter_ii_tl 41, 595
\l__enumext_counter_iii_tl 41, 596
\l__enumext_counter_iv_tl 41, 597
\c__enumext_counter_style_tl 33, 46, 227
\g__enumext_counter_styles_tl . 28, 41, 63, 605,
 623
\l__enumext_counter_v_tl 41, 598, 859
\l__enumext_counter_vi_tl 41, 599
\l__enumext_counter_vii_tl 41, 600, 791
\l__enumext_counter_viii_tl 41, 601, 807
\l__enumext_current_widest_dim 28, 63, 629, 725,
 771, 840, 844
__enumext_def_meta_key:nnn .. 134, 5364, 5392,
 5398, 5412
__enumext_default_item:n ... 3234, 3234, 3298
__enumext_define_counters:Nn 28, 585, 585, 594,
 595, 596, 597, 598, 599, 600, 601
__enumext_endminipage: . 36, 375, 384, 579, 4557,
 4853, 5129
\g__enumext_envir_name_tl 34, 30, 284, 299, 358,
 2082, 2087, 2097, 2237, 2242, 2247, 2261, 2266, 2271
\l__enumext_envir_name_tl . 33, 34, 30, 253, 263,
 312, 320, 328, 5760, 5763, 5770, 5773, 5780, 5783,
 5790, 5793, 5799, 5803, 5809, 5813, 5870, 5874
__enumext_execute_after_env: 35, 69, 72, 83, 87,
 3014, 3014, 3832, 4886
__enumext_fake_item_indent: . 1051, 1051, 3597
\l__enumext_fake_item_indent_v_dim 1070, 1075
\l__enumext_fake_item_indent_v_tl 1072, 3438,
 3442, 3450
__enumext_fake_item_indent_vii: . 1051, 1080,
 3630
\l__enumext_fake_item_indent_vii_dim . 1083,
 1087
\l__enumext_fake_item_indent_vii_tl .. 1085,
 4848
__enumext_fake_item_indent_viii: 1051, 1092,
 3635
\l__enumext_fake_item_indent_viii_dim 1095,
 1099
\l__enumext_fake_item_indent_viii_tl . 1097,
 5120
\l__enumext_fake_item_indent_X_tl 96
__enumext_fake_make_label_vii:n . 122, 4783,
 4783, 4840
__enumext_fake_make_label_viii:n 5057, 5075,
 5112
__enumext_filter_first_level:n .. 132, 5261,
 5261, 5295, 5306
__enumext_filter_first_level_key:n 132, 5261,
 5266, 5270
__enumext_filter_first_level_pair:nn . 132,
 5261, 5267, 5279
__enumext_filter_save_key:n .. 75, 2359, 2367,
 2390, 2396, 2398, 2398, 5192, 5196, 5200, 5204, 5208,
 5212
__enumext_filter_save_key_key:n .. 76, 2398,
 2403, 2407
__enumext_filter_save_key_pair:nn 76, 2398,
 2404, 2415
__enumext_filter_series:n 64, 1839, 1839, 1877,
 1889, 1894
__enumext_filter_series_key:n 64, 1839, 1844,
 1848
__enumext_filter_series_pair:nn .. 64, 1839,
 1845, 1856
__enumext_first_item_tmp_vii: 118, 120, 4639,
 4712, 4712
__enumext_first_item_tmp_viii: .. 126, 4901,
 4955, 4955
\g__enumext_footnote_standar_arg_seq .. 169,
 453, 464, 467
\g__enumext_footnote_standar_int 169, 447, 450,
 452, 455
\g__enumext_footnote_standar_int_seq .. 169,
 455, 460, 463, 468
\g__enumext_footnote_starred_arg_seq .. 169,

```

483, 494, 497  
 \g\_\_enumext\_footnote\_starred\_int 169, 477, 480, 482, 485  
 \g\_\_enumext\_footnote\_starred\_int\_seq .. 169, 485, 490, 493, 498  
 \\_\_enumext\_footnotes\_key\_bool ..... 36  
 \l\_\_enumext\_footnotes\_key\_bool 31, 36, 159, 396, 401, 408, 509, 525, 539, 552  
 \\_\_enumext\_footnotetext:nn .. 436, 436, 465, 495  
 \\_\_enumext\_foreach\_add\_body:n . 135, 5413, 5473, 5476  
 \l\_\_enumext\_foreach\_after\_tl ..... 5417, 5485  
 \l\_\_enumext\_foreach\_before\_tl .... 5415, 5480  
 \g\_\_enumext\_foreach\_default\_keys\_tl 134, 122, 5435, 5456  
 \\_\_enumext\_foreach\_keyans:nn .. 135, 5413, 5452, 5454  
 \l\_\_enumext\_foreach\_name\_prop\_tl . 122, 5458, 5483  
 \l\_\_enumext\_foreach\_print\_seq 122, 5468, 5474, 5478  
 \l\_\_enumext\_foreach\_sep\_tl ..... 5427, 5474  
 \l\_\_enumext\_foreach\_start\_int .... 5419, 5470  
 \l\_\_enumext\_foreach\_step\_int ..... 5423, 5471  
 \l\_\_enumext\_foreach\_stop\_int . 5421, 5463, 5465, 5472  
 \\_\_enumext\_foreach\_wrapper:n ..... 5425, 5481  
 \\_\_enumext\_getkeyans:nn .. 130, 5161, 5175, 5179  
 \\_\_enumext\_getkeyans\_aux:n 129, 5161, 5163, 5166  
 \l\_\_enumext\_hyperref\_bool 31, 36, 159, 393, 411, 428, 2645, 3137, 4792  
 \\_\_enumext\_hypertarget:nn 36, 386, 413, 417, 433  
 \\_\_enumext\_if\_is\_int:n ..... 219  
 \\_\_enumext\_if\_is\_int:nTF ..... 219, 877, 891  
 \\_\_enumext\_internal\_mini\_page: 39, 100, 119, 566, 566, 3645, 4652  
 \\_\_enumext\_is\_not\_nested: . 28, 33, 100, 119, 247, 247, 3644, 4651  
 \\_\_enumext\_is\_on\_first\_level: . 28, 34, 100, 119, 247, 273, 3651, 4664  
 \g\_\_enumext\_item\_anskey\_int 82, 90, 149, 345, 372, 373, 2222, 2594, 3151  
 \\_\_enumext\_item\_answer\_diff: 72, 87, 2218, 2218, 3021  
 \g\_\_enumext\_item\_answer\_diff\_int 72, 149, 346, 2220, 2227, 2251  
 \l\_\_enumext\_item\_column\_pos\_vii\_int 121, 4413, 4419, 4425, 4429, 4436, 4723, 4859, 4862  
 \l\_\_enumext\_item\_column\_pos\_viii\_int .. 126, 4462, 4468, 4474, 4478, 4485, 4966, 5135, 5138  
 \l\_\_enumext\_item\_column\_pos\_X\_int ..... 175  
 \g\_\_enumext\_item\_count\_all\_vii\_int 121, 4437, 4724, 4873, 4881  
 \g\_\_enumext\_item\_count\_all\_viii\_int 126, 4486, 4967, 5149, 5158  
 \g\_\_enumext\_item\_count\_all\_X\_int ..... 175  
 \g\_\_enumext\_item\_number\_bool ..... 149  
 \l\_\_enumext\_item\_number\_bool 71, 157, 2172, 2177, 2181, 2185, 2198, 2769, 2823, 3241, 3275, 4801  
 \g\_\_enumext\_item\_number\_int .. 71, 149, 344, 371, 373, 2171, 2176, 2180, 2184, 2197, 2222, 3240, 3274, 4800  
 \\_\_enumext\_item\_peek\_args\_vii: 121, 4720, 4725, 4727  
 \\_\_enumext\_item\_peek\_args\_viii: .. 126, 4963, 4968, 4970  
 \\_\_enumext\_item\_star\_exec: 93, 3253, 3280, 3325, 3346  
 \l\_\_enumext\_item\_star\_exec\_int 149, 3464, 3471, 3498, 4986, 5001, 5063  
 \l\_\_enumext\_item\_star\_wrap\_bool ..... 149  
 \l\_\_enumext\_item\_starred\_vii\_bool 4742, 4756, 4805  
 \l\_\_enumext\_item\_starred\_viii\_bool 4985, 5000, 5085, 5121  
 \l\_\_enumext\_item\_starred\_X\_bool ..... 175  
 \\_\_enumext\_item\_std:w . 36, 92, 96, 375, 379, 3244, 3250, 3278, 3438, 3442, 3450  
 \g\_\_enumext\_item\_symbol\_aux\_tl . 92, 126, 3258, 3261, 3286, 3330, 3350  
 \g\_\_enumext\_item\_symbol\_aux\_vii\_tl 4764, 4807, 4810, 4814, 4816  
 \g\_\_enumext\_item\_symbol\_aux\_X\_tl ..... 175  
 \l\_\_enumext\_item\_symbol\_sep\_vii\_dim .. 4772, 4779, 4813, 4815  
 \l\_\_enumext\_item\_symbol\_vii\_tl ..... 4810  
 \l\_\_enumext\_item\_text\_vii\_box .... 4832, 4857  
 \l\_\_enumext\_item\_text\_viii\_box ... 5104, 5133  
 \l\_\_enumext\_item\_text\_X\_box ..... 175  
 \l\_\_enumext\_item\_width\_vii\_dim ... 4352, 4361, 4440, 4448, 4449  
 \l\_\_enumext\_item\_width\_viii\_dim .. 4383, 4392, 4489, 4497, 4498  
 \l\_\_enumext\_item\_width\_X\_dim ..... 175  
 \l\_\_enumext\_itemindent\_X\_dim ..... 67  
 \l\_\_enumext\_itemsep\_i\_skip ... 1366, 1373, 1376, 1378, 1385, 1389, 1392, 1394, 1534, 1541, 1543, 1544, 1549, 1553, 1555, 1556  
 \l\_\_enumext\_itemsep\_ii\_skip .. 1406, 1413, 1416, 1418, 1425, 1429, 1432, 1434  
 \l\_\_enumext\_itemsep\_iii\_skip . 1445, 1452, 1455, 1457, 1464, 1468, 1471, 1473  
 \l\_\_enumext\_itemsep\_vii\_skip ..... 4879  
 \l\_\_enumext\_itemsep\_viii\_skip ..... 5156  
 \l\_\_enumext\_joined\_item\_aux\_vii\_int .. 4434, 4435, 4436, 4437, 4443  
 \l\_\_enumext\_joined\_item\_aux\_viii\_int . 4483, 4484, 4485, 4486, 4492  
 \l\_\_enumext\_joined\_item\_aux\_X\_int .... 175  
 \\_\_enumext\_joined\_item\_vii:w .. 121, 4720, 4730, 4731, 4733  
 \l\_\_enumext\_joined\_item\_vii\_int .. 4405, 4406, 4409, 4411, 4417, 4422, 4427, 4432, 4434, 4440  
 \\_\_enumext\_joined\_item\_viii:w . 126, 4963, 4973, 4974, 4976  
 \l\_\_enumext\_joined\_item\_viii\_int . 4454, 4455, 4458, 4460, 4466, 4471, 4476, 4481, 4483, 4489  
 \l\_\_enumext\_joined\_item\_X\_int ..... 175  
 \l\_\_enumext\_joined\_width\_vii\_dim . 4438, 4445, 4448, 4834, 4842  
 \l\_\_enumext\_joined\_width\_viii\_dim 4487, 4494, 4497, 5106, 5114  
 \l\_\_enumext\_joined\_width\_X\_dim ..... 175  
 \\_\_enumext\_keyans\_addto\_prop:n 88, 3034, 3034, 3453, 4244  
 \\_\_enumext\_keyans\_addto\_seq:n . 89, 3110, 3110, 3455, 4246  
 \\_\_enumext\_keyans\_addto\_seq\_link: 3110, 3131,



3133, 5035  
 \\_\_enumext\_keyans\_default\_item:n .. 96, 3433, 3433, 3472  
 \l\_\_enumext\_keyans\_env\_bool 30, 3676, 3689, 3841, 3931  
 \\_\_enumext\_keyans\_fake\_item\_indent: .. 1051, 1067, 3587  
 \l\_\_enumext\_keyans\_level\_h\_int .. 125, 24, 800, 826, 2753, 2815, 3088, 4658, 4918, 4919  
 \l\_\_enumext\_keyans\_level\_int .. 24, 1654, 2749, 2811, 3083, 3840, 3845, 4210  
 \\_\_enumext\_keyans\_make\_label: . 97, 3476, 3476, 3585  
 \\_\_enumext\_keyans\_make\_label\_box: 3476, 3480, 3485, 3520  
 \\_\_enumext\_keyans\_make\_label\_std: 3476, 3488, 3510  
 \\_\_enumext\_keyans\_mini\_right\_cmd:n 61, 1656, 1689, 1689  
 \\_\_enumext\_keyans\_mini\_set\_vskip: ..... 57  
 \\_\_enumext\_keyans\_minipage\_add\_space: 1485, 1511, 3870  
 \\_\_enumext\_keyans\_minipage\_set\_skip: . 1485, 1485, 1513  
 \\_\_enumext\_keyans\_multi\_addvspace: 1285, 1296, 3894  
 \\_\_enumext\_keyans\_multi\_set\_vskip: 54, 1285, 1285, 1298  
 \\_\_enumext\_keyans\_multicols\_start: 3858, 3873, 3875  
 \\_\_enumext\_keyans\_multicols\_stop: 1693, 3858, 3900, 3929  
 \\_\_enumext\_keyans\_name\_and\_start: 28, 34, 125, 306, 306, 3842, 4099, 4923  
 \\_\_enumext\_keyans\_parse\_keys:n 3854, 3854, 3949  
 \\_\_enumext\_keyans\_pic\_arg\_two: 109, 4092, 4115, 4145  
 \l\_\_enumext\_keyans\_pic\_level\_int .. 24, 1635, 2757, 2819, 3037, 3078, 3113, 3201, 4094, 4095  
 \\_\_enumext\_keyans\_pic\_parse\_keys:n 4092, 4101, 4144  
 \g\_\_enumext\_keyans\_pic\_parsep\_skip ... 138  
 \\_\_enumext\_keyans\_pic\_safe\_exec: . 109, 4092, 4092, 4143  
 \\_\_enumext\_keyans\_pic\_skip\_abs:N . 109, 4092, 4108, 4119  
 \\_\_enumext\_keyans\_pre\_itemsep\_skip: .. 1485, 1504, 1531  
 \\_\_enumext\_keyans\_redefine\_item: .. 96, 3458, 3458, 3584  
 \\_\_enumext\_keyans\_ref: ..... 45, 851, 868, 3586  
 \\_\_enumext\_keyans\_ref:n ..... 45, 848, 851, 851  
 \\_\_enumext\_keyans\_safe\_exec: . 3834, 3834, 3948  
 \\_\_enumext\_keyans\_set\_item\_width: 106, 3935, 3935, 3957  
 \\_\_enumext\_keyans\_show\_ans: .. 3154, 3162, 3181  
 \\_\_enumext\_keyans\_show\_item\_opt: .. 96, 3154, 3169, 3451, 4259, 5123  
 \\_\_enumext\_keyans\_show\_left:n . 96, 3154, 3154, 3448, 4253  
 \\_\_enumext\_keyans\_show\_pos: .. 3154, 3166, 3194  
 \\_\_enumext\_keyans\_starred\_item:n .. 96, 3445, 3445, 3467  
 \\_\_enumext\_keyans\_store\_ref: .. 88, 3057, 3057, 3454, 4245, 5033  
 \\_\_enumext\_keyans\_store\_ref\_aux\_i: 89, 3057, 3069, 3072  
 \\_\_enumext\_keyans\_store\_ref\_aux\_ii: 89, 3057, 3098, 3100  
 \\_\_enumext\_keyans\_unknown\_keys:n . 3371, 3375, 3379, 4090  
 \\_\_enumext\_keyans\_unknown\_keys:nn 3371, 3381, 3383  
 \\_\_enumext\_keyans\_wrapper\_label:n . 3476, 3492, 3516, 3529  
 \\_\_enumext\_keyans\_wrapper\_label\_viii:n 5057, 5092  
 \\_\_enumext\_keyans\_wrapper\_item:n . 2303, 3499, 3502, 5064, 5067  
 \\_\_enumext\_keyans\_wrapper\_opt:n .. 2300, 3177  
 \l\_\_enumext\_label\_copy\_i\_tl .. 2560, 3076, 3081, 3086, 3091  
 \l\_\_enumext\_label\_copy\_v\_tl ..... 3086  
 \l\_\_enumext\_label\_copy\_vi\_tl ..... 3081  
 \l\_\_enumext\_label\_copy\_vii\_tl 2536, 2547, 2576, 3076  
 \l\_\_enumext\_label\_copy\_viii\_tl ..... 3091  
 \l\_\_enumext\_label\_copy\_X\_tl ..... 161  
 \l\_\_enumext\_label\_fill\_left\_v\_tl ..... 3514  
 \l\_\_enumext\_label\_fill\_left\_X\_tl ..... 96  
 \l\_\_enumext\_label\_fill\_right\_v\_tl .... 3517  
 \l\_\_enumext\_label\_fill\_right\_X\_tl ..... 96  
 \l\_\_enumext\_label\_font\_style\_v\_tl 3515, 3528, 4257  
 \l\_\_enumext\_label\_font\_style\_vii\_tl ... 4820  
 \l\_\_enumext\_label\_font\_style\_viii\_tl .. 5091  
 \l\_\_enumext\_label\_i\_tl ..... 717  
 \l\_\_enumext\_label\_ii\_tl ..... 717  
 \l\_\_enumext\_label\_iii\_tl ..... 717  
 \l\_\_enumext\_label\_iv\_tl ..... 717  
 \\_\_enumext\_label\_style:Nnn 28, 41, 618, 618, 633, 722, 768, 837, 841  
 \l\_\_enumext\_label\_v\_tl 89, 834, 3042, 3118, 3188, 3228, 3447, 3452, 3952, 4123, 4252, 4254  
 \l\_\_enumext\_label\_vi\_tl 89, 834, 3039, 3115, 4252, 4254, 4258  
 \l\_\_enumext\_label\_vii\_tl . 763, 4751, 4774, 4781  
 \l\_\_enumext\_label\_viii\_tl 763, 4995, 5027, 5031  
 \l\_\_enumext\_label\_width\_by\_box .. 63, 614, 615  
 \\_\_enumext\_label\_width\_by\_box:Nn 41, 612, 612, 617, 629, 901  
 \l\_\_enumext\_labelsep\_i\_dim ... 3186, 3191, 3199, 3231, 5039, 5054  
 \l\_\_enumext\_labelsep\_v\_dim ..... 3884  
 \l\_\_enumext\_labelsep\_vii\_dim . 2661, 3186, 3199, 4347, 4357, 4441, 4716, 4772, 4827, 4836  
 \l\_\_enumext\_labelsep\_viii\_dim 4378, 4388, 4490, 4959, 5099, 5108  
 \l\_\_enumext\_labelwidth\_i\_dim . 3185, 3191, 3198, 3231, 5039, 5054  
 \l\_\_enumext\_labelwidth\_v\_dim ..... 3526, 3884  
 \l\_\_enumext\_labelwidth\_vii\_dim ... 2661, 3185, 3198, 4347, 4356, 4441, 4716, 4818, 4835  
 \l\_\_enumext\_labelwidth\_viii\_dim .. 4378, 4387, 4490, 4959, 5089, 5107  
 \l\_\_enumext\_leftmargin\_tmp\_v\_bool . 109, 4117  
 \l\_\_enumext\_leftmargin\_tmp\_X\_bool ..... 67  
 \l\_\_enumext\_leftmargin\_tmp\_X\_dim ..... 67



```

\l__enumext_leftmargin_X_dim 67
__enumext_level: 215, 215, 746, 749, 750, 758, 760,
 1054, 1058, 1062, 1130, 1134, 1138, 1142, 1225, 1227,
 1229, 1231, 1273, 1275, 1277, 1279, 1283, 1317, 1323,
 1328, 1330, 1333, 1336, 1349, 1352, 1663, 1667, 1673,
 1736, 1738, 1740, 1743, 1750, 1752, 1754, 1757, 2354,
 2356, 2358, 2386, 2387, 2389, 2445, 2453, 2457, 2461,
 2665, 2666, 3243, 3244, 3248, 3249, 3250, 3258, 3266,
 3267, 3270, 3277, 3278, 3282, 3285, 3287, 3321, 3322,
 3323, 3326, 3329, 3340, 3341, 3343, 3344, 3347, 3682,
 3695, 3702, 3710, 3713, 3715, 3717, 3718, 3719, 3720,
 3723, 3728, 3734, 3740, 3747, 3760, 3762, 3765, 3766,
 3768, 3772, 3778, 3803, 3808, 3819, 3821
\l__enumext_level_h_int 119, 24, 256, 279, 293, 784,
 819, 1642, 2168, 2188, 2555, 2789, 2801, 3690, 4653,
 4654
\l__enumext_level_int . 100, 24, 217, 266, 278, 294,
 568, 1237, 1362, 1641, 2162, 2194, 2532, 2542, 2548,
 2554, 2561, 2570, 2575, 2788, 2800, 3016, 3601, 3646,
 3647, 3658, 3666, 3680, 3693, 3724, 3849, 4206, 4696,
 4706, 4931, 5800, 5804, 5810, 5814
__enumext_list_arg_two_i: 3566
__enumext_list_arg_two_ii: 3566
__enumext_list_arg_two_iii: 3566
__enumext_list_arg_two_iv: 3566
__enumext_list_arg_two_v: . 96, 3566, 3954, 4118
__enumext_list_arg_two_vii: 3607, 4633
__enumext_list_arg_two_viii: 3607, 4895
\l__enumext_listoffset_v_dim . 3886, 3940, 3943
\l__enumext_listparindent_vii_dim 4843, 4847
\l__enumext_listparindent_viii_dim 5115, 5119
__enumext_log_answer_vars: . 35, 360, 368, 3023
__enumext_log_global_vars: . 35, 360, 360, 3022
__enumext_make_label: 93, 3301, 3301, 3595
__enumext_make_label_box: . . . 3301, 3305, 3310,
 3333
__enumext_make_label_std: . . . 3301, 3313, 3317
\l__enumext_mark_answer_sym_tl 78, 2308, 2511,
 2678, 3203, 3216, 5043
\l__enumext_mark_position_str 126, 2312, 2313,
 2339, 2340, 2509
\l__enumext_mark_ref_sym_tl . 2325, 2650, 3145
\l__enumext_meta_path_tl . 122, 5388, 5389, 5391,
 5392
\c__enumext_meta_paths_prop 133, 5364
__enumext_mini_addvspace_vii: 60, 1621, 1621,
 4515
__enumext_mini_addvspace_viii: 60, 1621, 1627,
 4580
__enumext_mini_env* 566
__enumext_mini_page 1673, 1700, 3772, 3871, 4517,
 4582, 4603
__enumext_mini_right_cmd:n . 60, 61, 1658, 1660,
 1660
__enumext_mini_set_vskip_vii: 59, 1564, 1564,
 1623
__enumext_mini_set_vskip_viii: 59, 1564, 1586,
 1629
__enumext_minipage:w 36, 375, 383, 573, 4540, 4842,
 5114
\l__enumext_minipage_active_v_bool 3868, 3891,
 3916
\g__enumext_minipage_active_vii_bool . 117,
 4529, 4538, 4560
\l__enumext_minipage_active_vii_bool . 4511,
 4522
\g__enumext_minipage_active_viii_bool 4593,
 4601, 4620
\l__enumext_minipage_active_viii_bool 4576,
 4587
\g__enumext_minipage_active_X_bool . . . 175
\l__enumext_minipage_active_X_bool 83
__enumext_minipage_add_space: . 55, 103, 1313,
 1339, 3770
\g__enumext_minipage_after_skip 83, 1568, 1580,
 4558, 4618
\l__enumext_minipage_after_skip . 55, 103, 83,
 1326, 1366, 1368, 1373, 1376, 1380, 1385, 1389, 1392,
 1396, 1408, 1413, 1416, 1420, 1425, 1429, 1432, 1436,
 1447, 1452, 1455, 1459, 1464, 1468, 1471, 1475, 1487,
 1501, 1534, 1536, 1541, 1543, 1545, 1549, 1553, 1555,
 1557, 1588, 1601, 1615, 1669, 1696, 3926
\g__enumext_minipage_center_vii_bool . 4544,
 4561
\g__enumext_minipage_center_viii_bool 4605,
 4621
\g__enumext_minipage_center_X_bool . . . 175
\l__enumext_minipage_hsep_v_dim 3866
\l__enumext_minipage_hsep_vii_dim 4509
\l__enumext_minipage_hsep_viii_dim . . . 4574
\l__enumext_minipage_left_skip 83, 1488, 1566,
 1571, 1575, 1589, 1593, 1607, 1625, 1631
\l__enumext_minipage_left_v_dim . 3864, 3871
\l__enumext_minipage_left_vii_dim 4505, 4517
\l__enumext_minipage_left_viii_dim 4570, 4582
\l__enumext_minipage_left_X_dim 83
\g__enumext_minipage_right_skip 83, 1567, 1572,
 1576, 4543, 4604
\l__enumext_minipage_right_skip . 55, 83, 1315,
 1321, 1326, 1328, 1330, 1489, 1490, 1496, 1501, 1502,
 1503, 1508, 1590, 1597, 1611, 1675, 1702
\l__enumext_minipage_right_v_dim . 1691, 1700,
 3862, 3866
\g__enumext_minipage_right_vii_dim 116, 4513,
 4540, 4563
\l__enumext_minipage_right_vii_dim 116, 4503,
 4508, 4514
\g__enumext_minipage_right_viii_dim . 4578,
 4603, 4623
\l__enumext_minipage_right_viii_dim . 4568,
 4573, 4579
\g__enumext_minipage_right_X_dim 175
\g__enumext_minipage_right_X_skip 175
__enumext_minipage_set_skip: . 55, 1313, 1313,
 1341
\g__enumext_minipage_stat_int . 103, 83, 1680,
 1707, 3769, 3780, 3785, 3869, 3918, 3923
\l__enumext_minipage_temp_skip 83, 1387, 1397,
 1400, 1427, 1437, 1440, 1466, 1476, 1479, 1551, 1558,
 1560
\l__enumext_miniright_code_vii_box 4551, 4555
\g__enumext_miniright_code_vii_tl 117, 4546,
 4553, 4562
\l__enumext_miniright_code_viii_box . 4612,
 4616
\g__enumext_miniright_code_viii_tl 4607, 4614,
 4622
\l__enumext_miniright_code_X_box 175
\l__enumext_mode_box_bool 638, 3308, 3483

```

`\__enumext_multi_addvspace:` [53](#), [102](#), [1268](#), [1268](#), [3731](#)  
`\__enumext_multi_set_vskip:` [53](#), [1223](#), [1223](#), [1270](#)  
`\l__enumext_multicols_above_ii_skip` ... [1242](#)  
`\l__enumext_multicols_above_iii_skip` .. [1251](#)  
`\l__enumext_multicols_above_iv_skip` ... [1260](#)  
`\l__enumext_multicols_above_v_skip` [1287](#), [1301](#), [1311](#), [1502](#)  
`\l__enumext_multicols_above_X_skip` .... [75](#)  
`\l__enumext_multicols_below_ii_skip` .. [1369](#), [1378](#), [1382](#), [1394](#), [1399](#)  
`\l__enumext_multicols_below_iii_skip` . [1409](#), [1418](#), [1422](#), [1434](#), [1439](#)  
`\l__enumext_multicols_below_iv_skip` .. [1448](#), [1457](#), [1461](#), [1473](#), [1478](#)  
`\l__enumext_multicols_below_v_skip` [1291](#), [1305](#), [1503](#), [1537](#), [1544](#), [1546](#), [1556](#), [1559](#), [3908](#)  
`\l__enumext_multicols_below_X_skip` .... [75](#)  
`\g__enumext_multicols_right_X_skip` .... [75](#)  
`\__enumext_multicols_start:` [102](#), [103](#), [3707](#), [3707](#), [3774](#)  
`\__enumext_multicols_stop:` [102](#), [1665](#), [3737](#), [3737](#), [3790](#)  
`\__enumext_nested_base_line_fix:` [48](#), [100](#), [975](#), [981](#), [3662](#)  
`\__enumext_newlabel:nn` [31](#), [37](#), [79](#), [421](#), [421](#), [2586](#), [3104](#)  
`\l__enumext_newlabel_arg_one_tl` [31](#), [37](#), [79](#), [89](#), [161](#), [2579](#), [2587](#), [2649](#), [3093](#), [3105](#), [3143](#)  
`\l__enumext_newlabel_arg_two_tl` [31](#), [37](#), [78](#), [161](#), [2535](#), [2545](#), [2558](#), [2573](#), [2588](#), [3080](#), [3085](#), [3090](#), [3106](#)  
`\__enumext_parse_foreach_keys:n` .. [5413](#), [5429](#), [5446](#)  
`\__enumext_parse_foreach_keys:nn` . [5413](#), [5436](#), [5448](#)  
`\__enumext_parse_keys:n` [48](#), [64](#), [3653](#), [3653](#), [3815](#)  
`\__enumext_parse_keys_vii:n` [64](#), [4628](#), [4666](#), [4666](#)  
`\__enumext_parse_keys_viii:n` . [4891](#), [4936](#), [4936](#)  
`\__enumext_parse_save_key:n` [75](#), [2379](#), [2384](#), [2384](#)  
`\__enumext_parse_save_key_vii:n` [75](#), [2374](#), [2384](#), [2392](#)  
`\__enumext_parse_series:n` .. [64](#), [100](#), [119](#), [1865](#), [1865](#), [3661](#), [4672](#)  
`\__enumext_parse_store_keys:n`..... [100](#)  
`\l__enumext_parsep_i_skip` ..... [1240](#), [1244](#)  
`\l__enumext_parsep_ii_skip` ..... [1249](#), [1253](#)  
`\l__enumext_parsep_iii_skip` ..... [1258](#), [1262](#)  
`\l__enumext_parsep_vii_skip` ..... [4844](#)  
`\l__enumext_parsep_viii_skip` ..... [5116](#)  
`\l__enumext_partopsep_v_skip` . [1303](#), [1307](#), [1498](#), [1521](#)  
`\l__enumext_partopsep_viii_skip` ..... [1599](#)  
`\__enumext_phantomsection:` [36](#), [386](#), [414](#), [418](#), [434](#)  
`\__enumext_pre_itemsep_skip:` [55](#), [56](#), [1331](#), [1360](#), [1360](#)  
`\__enumext_print_footnote:` .. [436](#), [458](#), [522](#), [527](#)  
`\__enumext_print_footnote_mini:` [436](#), [488](#), [549](#), [554](#)  
`\__enumext_print_footnote_standar:` [500](#), [516](#), [580](#)  
`\__enumext_print_footnote_starred:` [500](#), [545](#), [560](#), [564](#)  
`\__enumext_print_keyans_box:NN` [78](#), [2503](#), [2503](#), [2516](#), [2660](#), [2664](#), [3190](#), [3230](#), [5039](#), [5054](#)  
`\l__enumext_print_keyans_i_tl` .... [5197](#), [5219](#)  
`\l__enumext_print_keyans_ii_tl` ... [5201](#), [5220](#)  
`\l__enumext_print_keyans_iii_tl` .. [5205](#), [5221](#)  
`\l__enumext_print_keyans_iv_tl` ... [5209](#), [5222](#)  
`\l__enumext_print_keyans_star_bool` . [48](#), [131](#), [126](#), [987](#), [995](#), [5242](#), [5247](#)  
`\l__enumext_print_keyans_starred_tl` [130](#), [131](#), [126](#), [5193](#), [5240](#)  
`\l__enumext_print_keyans_vii_tl` [130](#), [5213](#), [5223](#)  
`\l__enumext_print_keyans_X_tl` ..... [126](#)  
`\__enumext_printkeyans:nnn` [131](#), [5216](#), [5224](#), [5227](#)  
`\__enumext_redefine_item:` . [93](#), [3290](#), [3290](#), [3594](#)  
`\l__enumext_ref_key_arg_tl` [43](#), [46](#), [230](#), [739](#), [740](#), [752](#), [783](#), [786](#), [796](#), [802](#), [812](#), [853](#), [854](#), [864](#)  
`\l__enumext_ref_the_count_tl` . [43](#), [46](#), [746](#), [749](#), [752](#), [791](#), [793](#), [796](#), [807](#), [809](#), [812](#), [859](#), [861](#), [864](#)  
`\__enumext_regex_counter_style:` .. [33](#), [43](#), [225](#), [225](#), [747](#), [792](#), [808](#), [860](#)  
`\__enumext_register_counter_counter_style:Nn` .. [602](#), [602](#), [607](#), [608](#), [609](#), [610](#), [611](#)  
`\__enumext_remove_extra_parsep_vii:` .. [4646](#), [4868](#), [4868](#)  
`\__enumext_remove_extra_parsep_viii:` . [4908](#), [5144](#), [5144](#)  
`\__enumext_renew_footnote:` .. [436](#), [440](#), [506](#), [511](#)  
`\__enumext_renew_footnote_mini:` [436](#), [470](#), [536](#), [541](#)  
`\__enumext_renew_footnote_standar:` [500](#), [500](#), [572](#)  
`\__enumext_renew_footnote_starred:` [500](#), [532](#), [4838](#), [5110](#)  
`\l__enumext_renew_the_count_v_tl` [862](#), [870](#), [872](#)  
`\l__enumext_renew_the_count_vii_tl` [794](#), [821](#), [823](#)  
`\l__enumext_renew_the_count_viii_tl` [810](#), [828](#), [830](#)  
`\l__enumext_renew_the_count_X_tl` ..... [46](#)  
`\__enumext_rescan_anskey_env:n` .. [85](#), [87](#), [2832](#), [2898](#), [2993](#), [3001](#)  
`\__enumext_reset_global_bool:` .. [336](#), [339](#), [348](#)  
`\__enumext_reset_global_int:` ... [336](#), [338](#), [342](#)  
`\__enumext_reset_global_tl:` .... [336](#), [340](#), [354](#)  
`\__enumext_reset_global_vars:` . [35](#), [87](#), [336](#), [336](#), [3031](#)  
`\l__enumext_resume_active_bool` [64](#), [67](#), [57](#), [1869](#), [1989](#)  
`\__enumext_resume_counter:` .. [66](#), [67](#), [1987](#), [1993](#), [2000](#)  
`\__enumext_resume_counter:n` . [64](#), [67](#), [1958](#), [1963](#), [1987](#), [1987](#), [2057](#), [2065](#)  
`\__enumext_resume_counter_save_ans:` .. [67](#), [68](#), [1987](#), [1998](#), [2030](#)  
`\__enumext_resume_counter_series:` . [67](#), [1987](#), [1996](#), [2013](#)  
`\g__enumext_resume_int` ... [57](#), [1910](#), [2004](#), [2005](#)  
`\__enumext_resume_last:n` [64](#), [65](#), [1865](#), [1871](#), [1884](#)  
`\l__enumext_resume_name_tl` [57](#), [1906](#), [1914](#), [1917](#), [1933](#), [1941](#), [1944](#), [1990](#), [1991](#), [2019](#), [2026](#)  
`\__enumext_resume_save_counter:` . [65](#), [103](#), [120](#), [1897](#), [1897](#), [3796](#), [4690](#)  
`\__enumext_resume_series:n` . [66](#), [1833](#), [1954](#), [1954](#)  
`\__enumext_resume_starred:` . [68](#), [1834](#), [2051](#), [2051](#)  
`\g__enumext_resume_vii_int` [57](#), [1937](#), [2009](#), [2010](#)  
`\l__enumext_rightmargin_vii_dim` .. [4359](#), [4363](#)

4368  
 \l\_\_enumext\_rightmargin\_viii\_dim . 4390, 4394,  
 4399  
 \\_\_enumext\_safe\_exec: . . 39, 100, 3642, 3642, 3814  
 \\_\_enumext\_safe\_exec\_vii: . 39, 4627, 4649, 4649  
 \\_\_enumext\_safe\_exec\_viii: 125, 4890, 4912, 4912  
 \\_\_enumext\_second\_part: . . 103, 3776, 3776, 3828  
 \\_\_enumext\_second\_part\_v: . . . 3858, 3914, 3962  
 \l\_\_enumext\_series\_name\_tl . . . . . 67  
 \l\_\_enumext\_series\_str . 65, 100, 119, 1831, 1867,  
 1875, 1876, 1878, 1880, 1901, 1904, 1908, 1928, 1931,  
 1935, 3657, 4670  
 \\_\_enumext\_set\_error:nn . . . . . 5323, 5360, 5362  
 \\_\_enumext\_set\_item\_width: 103, 3798, 3798, 3824  
 \\_\_enumext\_set\_parse:n . . . . . 5323, 5334, 5350  
 \l\_\_enumext\_setkey\_tmpa\_int . . . 117, 5327, 5331  
 \l\_\_enumext\_setkey\_tmpa\_seq . . 117, 5325, 5335,  
 5341, 5343, 5345, 5357  
 \l\_\_enumext\_setkey\_tmpa\_tl . . . 117, 5333, 5337  
 \l\_\_enumext\_setkey\_tmpb\_seq . . 117, 5326, 5329,  
 5333, 5334  
 \l\_\_enumext\_setkey\_tmpb\_tl 117, 5352, 5354, 5355  
 \l\_\_enumext\_show\_answer\_bool . 2319, 2343, 2672,  
 3160, 3174, 3497, 4249, 5037, 5062  
 \\_\_enumext\_show\_length:nnn . . 50, 233, 233, 5571,  
 5572, 5573, 5574, 5575, 5576, 5577, 5578, 5579, 5580,  
 5586, 5587, 5588, 5589, 5590, 5591, 5592, 5593, 5594,  
 5595  
 \l\_\_enumext\_show\_position\_bool . . . 2322, 2346,  
 2676, 3164, 3175, 4250, 5041  
 \g\_\_enumext\_standar\_bool 33, 100, 30, 255, 258, 277,  
 351, 502, 518, 1899, 1964, 1976, 2002, 2015, 2053,  
 2193, 2207, 2540, 2553, 2568, 3677  
 \l\_\_enumext\_standar\_bool 100, 103, 30, 1649, 2541,  
 3649, 3795, 4663  
 \l\_\_enumext\_standar\_first\_bool 34, 100, 30, 282,  
 1886, 2033, 2095, 2102  
 \\_\_enumext\_standar\_item\_vii:w . 121, 4720, 4738,  
 4740  
 \\_\_enumext\_standar\_item\_viii:w 126, 4963, 4981,  
 4983  
 \\_\_enumext\_standar\_ref: . . . . 43, 737, 756, 3596  
 \\_\_enumext\_standar\_ref:n . . . . 43, 729, 737, 737  
 \g\_\_enumext\_standar\_series\_tl . 57, 1888, 1889,  
 2055, 2058  
 \\_\_enumext\_standar\_unknown\_keys:n 3411, 3415,  
 3419  
 \\_\_enumext\_standar\_unknown\_keys:nn 3411, 3421,  
 3423  
 \g\_\_enumext\_starred\_bool 33, 119, 30, 265, 268, 292,  
 352, 1648, 1926, 1969, 1980, 2007, 2022, 2061, 2167,  
 2213, 2531, 3074, 4564  
 \l\_\_enumext\_starred\_bool 119, 120, 125, 30, 2569,  
 2604, 2610, 2658, 2947, 2952, 3183, 3196, 3650, 4662,  
 4689, 4924, 4928  
 \\_\_enumext\_starred\_columns\_set\_vii: . . 4341,  
 4341, 4637  
 \\_\_enumext\_starred\_columns\_set\_viii: . 4341,  
 4372, 4899  
 \l\_\_enumext\_starred\_first\_bool 34, 119, 30, 297,  
 985, 994, 1891, 2042, 2095, 2102  
 \\_\_enumext\_starred\_item:nn . . . 3253, 3253, 3296  
 \\_\_enumext\_starred\_item\_exec: . 127, 4998, 5029,  
 5087  
 \\_\_enumext\_starred\_item\_vii:w . 121, 4720, 4737,  
 4754  
 \\_\_enumext\_starred\_item\_vii\_aux\_i:w . . 4720,  
 4759, 4762  
 \\_\_enumext\_starred\_item\_vii\_aux\_ii:w . 4720,  
 4760, 4765, 4767  
 \\_\_enumext\_starred\_item\_vii\_aux\_iii:w 4720,  
 4770, 4777  
 \\_\_enumext\_starred\_item\_viii:w 126, 127, 4980,  
 4998, 4998  
 \\_\_enumext\_starred\_item\_viii\_aux\_i:w . . 127,  
 4998, 5004, 5007  
 \\_\_enumext\_starred\_item\_viii\_aux\_ii:w . 127,  
 4998, 5005, 5022, 5024  
 \\_\_enumext\_starred\_joined\_item\_vii:n 115, 121,  
 4403, 4403, 4735  
 \\_\_enumext\_starred\_joined\_item\_viii:n . 115,  
 126, 4403, 4452, 4978  
 \\_\_enumext\_starred\_ref: . . . . 44, 781, 817, 3627  
 \\_\_enumext\_starred\_ref:n . . . . 44, 775, 781, 781  
 \g\_\_enumext\_starred\_series\_tl . 57, 1893, 1894,  
 2063, 2066  
 \\_\_enumext\_starred\_unknown\_keys:n 3393, 3395,  
 3397  
 \\_\_enumext\_starred\_unknown\_keys:nn 3393, 3399,  
 3401  
 \\_\_enumext\_start\_from:NNn 45, 875, 875, 888, 910,  
 916  
 \l\_\_enumext\_start\_i\_int . . . . . 2005, 2017, 2036  
 \\_\_enumext\_start\_item\_tmp\_vii: 118, 4640, 4720,  
 4720  
 \\_\_enumext\_start\_item\_tmp\_viii: . . 4902, 4963,  
 4963  
 \\_\_enumext\_start\_item\_vii:w 121, 123, 4746, 4751,  
 4774, 4781, 4829, 4829  
 \\_\_enumext\_start\_item\_viii:w . . 126, 4990, 4995,  
 5027, 5101, 5101  
 \g\_\_enumext\_start\_line\_tl 34, 30, 285, 300, 357,  
 2237, 2242, 2247, 2261, 2266, 2271  
 \\_\_enumext\_start\_list:nn . 36, 98, 375, 377, 3818,  
 3951, 4631, 4893  
 \\_\_enumext\_start\_list\_tag:n . . 3964, 3990, 4839,  
 5111  
 \\_\_enumext\_start\_mini\_vii: 120, 4501, 4501, 4681  
 \\_\_enumext\_start\_mini\_viii: . . 125, 4566, 4566,  
 4947  
 \\_\_enumext\_start\_save\_ans\_msg: 69, 2079, 2079,  
 2104  
 \\_\_enumext\_start\_store\_level: . 101, 3671, 3671,  
 3817  
 \\_\_enumext\_start\_store\_level\_vii: 120, 4630,  
 4692, 4692  
 \l\_\_enumext\_start\_vii\_int . . . 2010, 2024, 2045  
 \l\_\_enumext\_start\_X\_int . . . . . 96  
 \\_\_enumext\_stop\_item\_tmp\_vii: 118, 120, 121, 123,  
 4639, 4645, 4722, 4831  
 \\_\_enumext\_stop\_item\_tmp\_viii: 126, 4901, 4907,  
 4965, 5103  
 \\_\_enumext\_stop\_item\_vii: 123, 4829, 4831, 4851  
 \\_\_enumext\_stop\_item\_viii: . . . 5101, 5103, 5127  
 \\_\_enumext\_stop\_list: 36, 117, 120, 375, 378, 3742,  
 3750, 3904, 3911, 4524, 4532, 4589, 4596  
 \\_\_enumext\_stop\_list\_tag:n . . . 3964, 4006, 4854,  
 5130

```

__enumext_stop_mini_vii: 117, 120, 4501, 4520,
 4685
__enumext_stop_mini_viii: 125, 4566, 4585, 4951
__enumext_stop_save_ans_msg: . 69, 2079, 2084,
 3020
__enumext_stop_start_list_tag: .. 3964, 3998,
 4841, 5113
__enumext_stop_store_level: .. 101, 102, 3700,
 3700, 3743, 3751
__enumext_stop_store_level_vii: .. 117, 120,
 4525, 4533, 4692, 4702
\l__enumext_store_active_bool 30, 69, 108, 2034,
 2043, 2111, 2745, 3675, 3688, 3836, 3844, 4202, 4694,
 4704, 4914, 4930
__enumext_store_active_keys:n 74, 75, 100, 2352,
 2352, 3668
__enumext_store_active_keys_vii:n 74, 75, 119,
 2352, 2362, 4673
__enumext_store_addto_prop:n 76, 88, 2427, 2427,
 2435, 2595, 3055, 5032
__enumext_store_addto_seq:n 76, 90, 2436, 2436,
 2440, 2447, 2461, 2469, 2478, 2492, 2500, 2653, 3148
\l__enumext_store_anskey_arg_tl 30, 79, 80, 108,
 2601, 2606, 2608, 2613, 2620, 2623, 2633, 2638, 2641,
 2647, 2653
__enumext_store_anskey_code:n 79, 82, 87, 2592,
 2592, 2738, 2991, 2999
\l__enumext_store_anskey_env_tl .. 30, 86, 108,
 2921, 2925, 2931, 2993, 3001
\l__enumext_store_anskey_opt_tl .. 30, 86, 108,
 2922, 2949, 2955, 2962, 2968, 2978, 2988, 2997
__enumext_store_anskey_safe_outer: 82
\g__enumext_store_columns_break_bool . 2845,
 2946, 3008
\l__enumext_store_columns_break_bool . 2603,
 2694
\l__enumext_store_current_label_tl 30, 88-90,
 127, 108, 3036, 3039, 3042, 3048, 3053, 3055, 3112,
 3115, 3118, 3124, 3129, 3139, 3148, 5009, 5014, 5018,
 5031, 5032, 5034
\l__enumext_store_current_label_tmp_tl . 30,
 108, 3447, 3452
\l__enumext_store_current_opt_arg_tl 30, 127,
 108, 3158, 3171, 3177, 5020
__enumext_store_internal_ref: .. 78, 79, 2517,
 2517, 2598
\g__enumext_store_item_join_int .. 2848, 2953,
 2957, 3009
\l__enumext_store_item_join_int .. 2611, 2615,
 2697
\g__enumext_store_item_star_bool . 2850, 2960,
 3010
\l__enumext_store_item_star_bool . 2618, 2699
\g__enumext_store_item_symbol_sep_dim 2855,
 2975, 2980, 3012
\l__enumext_store_item_symbol_sep_dim 2630,
 2635, 2704
\g__enumext_store_item_symbol_tl . 2853, 2966,
 2970, 3011
\l__enumext_store_item_symbol_tl . 2621, 2625,
 2702
\l__enumext_store_keyans_item_opt_sep_-
 tl 2305, 3046, 3050, 3122, 3126, 5012, 5016
__enumext_store_level_close: . 77, 2441, 2465,
 3704
__enumext_store_level_close_vii: . 77, 2472,
 2496, 4708
__enumext_store_level_open: 77, 101, 2441, 2441,
 3683, 3696
__enumext_store_level_open_vii: .. 77, 2472,
 2472, 4698
\g__enumext_store_name_tl 30, 69, 108, 356, 363,
 364, 365, 366, 2087, 2113, 2236, 2241, 2246, 2260,
 2265, 2270, 3018
\l__enumext_store_name_tl 30, 69, 71, 108, 1920,
 1923, 1947, 1950, 2038, 2047, 2082, 2091, 2092, 2113,
 2114, 2115, 2117, 2118, 2120, 2122, 2123, 2125, 2127,
 2128, 2152, 2429, 2431, 2438, 2581, 2582, 2684, 2927,
 3095, 3096, 3209, 3222, 5049
\l__enumext_store_ref_key_bool 79, 2328, 2596,
 2644, 3059, 3136
\l__enumext_store_save_key_vii_bool .. 2364,
 2394
\l__enumext_store_save_key_vii_tl 2366, 2367,
 2395, 2396, 2476, 2484, 2488, 2492
\l__enumext_store_save_key_X_bool .. 74, 126
\l__enumext_store_save_key_X_tl .. 74, 75, 126
\l__enumext_store_upper_level_X_bool .. 126
__enumext_storing_exec: 69, 84, 2089, 2105, 2109
__enumext_storing_set:n .. 69, 2074, 2089, 2089
\l__enumext_the_counter_v_tl 861
\l__enumext_the_counter_vii_tl 793
\l__enumext_the_counter_viii_tl 809
\l__enumext_the_counter_X_tl 46
__enumext_tmp:n 41, 45, 50, 56, 67, 74, 75, 82, 90, 95,
 96, 107, 130, 137, 164, 168, 175, 195, 634, 643, 1827,
 1838, 2070, 2078, 2131, 2149, 2290, 2333, 2334, 2351,
 2370, 2383, 2519, 2526, 2527, 2548, 2561, 2564, 2575,
 3061, 3068, 3371, 3378, 3411, 3418, 3566, 3606, 3607,
 3641
__enumext_tmp:nn 644, 665, 666, 700, 701, 716, 905,
 930, 1007, 1029, 1030, 1050, 1104, 1112, 1113, 1127,
 1192, 1208, 1209, 1222, 1716, 1732, 3355, 3370
__enumext_tmp:nnn 717, 733, 734, 735, 736, 763, 779,
 780
__enumext_tmp:nnnnn 931, 956, 959, 962, 964, 966,
 969, 972
__enumext_tmp:w 5172, 5175
\l__enumext_tmpa_vii_int 4351, 4354, 4363, 4394
\l__enumext_tmpa_viii_int 4382, 4385
\l__enumext_tmpa_X_dim 175
\l__enumext_tmpa_X_int 175
\l__enumext_topsep_v_skip 1289, 1293, 1492, 4195
\l__enumext_topsep_vii_skip .. 1569, 1578, 1582
\l__enumext_topsep_viii_skip . 1591, 1613, 1617
__enumext_undefine_anskey_env: . 83, 87, 2778,
 2778, 3029
__enumext_unskip_unkern: .. 33, 239, 239, 1342,
 1514, 3745, 3746, 3786, 3906, 3907, 3924, 4845, 4846,
 5117, 5118
\l__enumext_vspace_a_star_v_bool 1765
\l__enumext_vspace_a_star_vii_bool ... 1787
\l__enumext_vspace_a_star_viii_bool ... 1798
\l__enumext_vspace_a_star_X_bool 96
__enumext_vspace_above: 62, 102, 1733, 1733, 3756
__enumext_vspace_above_v: . 62, 1761, 1761, 3860
\l__enumext_vspace_above_v_skip .. 1763, 1767,
 1769
__enumext_vspace_above_vii: 63, 120, 1783, 1783,
 4678

```



|                                                  |                                                                                                                                                                             |
|--------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>\l_enumext_vspace_above_vii_skip</code>    | 1785, 1789, 1791                                                                                                                                                            |
| <code>\_enumext_vspace_above_viii:</code>        | 63, 1783, 1794, 4945                                                                                                                                                        |
| <code>\l_enumext_vspace_above_viii_skip</code>   | 1796, 1800, 1802                                                                                                                                                            |
| <code>\l_enumext_vspace_b_star_v_bool</code>     | 1776                                                                                                                                                                        |
| <code>\l_enumext_vspace_b_star_vii_bool</code>   | 1809                                                                                                                                                                        |
| <code>\l_enumext_vspace_b_star_viii_bool</code>  | 1820                                                                                                                                                                        |
| <code>\l_enumext_vspace_b_star_X_bool</code>     | 96                                                                                                                                                                          |
| <code>\_enumext_vspace_below:</code>             | 62, 103, 1747, 1747, 3794                                                                                                                                                   |
| <code>\_enumext_vspace_below_v:</code>           | 63, 1772, 1772, 3933                                                                                                                                                        |
| <code>\l_enumext_vspace_below_v_skip</code>      | 1774, 1778, 1780                                                                                                                                                            |
| <code>\_enumext_vspace_below_vii:</code>         | 63, 120, 1805, 1805, 4688                                                                                                                                                   |
| <code>\l_enumext_vspace_below_vii_skip</code>    | 1807, 1811, 1813                                                                                                                                                            |
| <code>\_enumext_vspace_below_viii:</code>        | 63, 1805, 1816, 4953                                                                                                                                                        |
| <code>\l_enumext_vspace_below_viii_skip</code>   | 1818, 1822, 1824                                                                                                                                                            |
| <code>\_enumext_widest_from:nNNn</code>          | 46, 889, 889, 904, 923                                                                                                                                                      |
| <code>\g_enumext_widest_label_tl</code>          | 28, 41, 63, 622, 626, 630                                                                                                                                                   |
| <code>\l_enumext_wrap_label_opt_v_bool</code>    | 3441                                                                                                                                                                        |
| <code>\l_enumext_wrap_label_opt_vii_bool</code>  | 121, 4745                                                                                                                                                                   |
| <code>\l_enumext_wrap_label_opt_viii_bool</code> | 126, 4989                                                                                                                                                                   |
| <code>\l_enumext_wrap_label_opt_X_bool</code>    | 96                                                                                                                                                                          |
| <code>\l_enumext_wrap_label_v_bool</code>        | 3437, 3441, 3449, 3496, 3504                                                                                                                                                |
| <code>\l_enumext_wrap_label_vii_bool</code>      | 121, 4745, 4749, 4757, 4821                                                                                                                                                 |
| <code>\l_enumext_wrap_label_viii_bool</code>     | 126, 4989, 4993, 5002, 5061, 5069, 5093                                                                                                                                     |
| <code>\l_enumext_wrap_label_X_bool</code>        | 96                                                                                                                                                                          |
| <code>\_enumext_wrapper_label_v:n</code>         | 3502, 3506, 4258                                                                                                                                                            |
| <code>\_enumext_wrapper_label_vii:n</code>       | 4823                                                                                                                                                                        |
| <code>\_enumext_wrapper_label_viii:n</code>      | 5067, 5071, 5095                                                                                                                                                            |
| <code>\l_enumext_write_aux_file_tl</code>        | 31, 79, 89, 161, 2584, 2590, 3102, 3108                                                                                                                                     |
| <code>enumext*</code>                            | 5, 4625                                                                                                                                                                     |
| <code>enumXi</code>                              | 585                                                                                                                                                                         |
| <code>enumXii</code>                             | 585                                                                                                                                                                         |
| <code>enumXiii</code>                            | 585                                                                                                                                                                         |
| <code>enumXiv</code>                             | 585                                                                                                                                                                         |
| <code>enumXv</code>                              | 585                                                                                                                                                                         |
| <code>enumXvi</code>                             | 585                                                                                                                                                                         |
| <code>enumXvii</code>                            | 585                                                                                                                                                                         |
| <code>enumXviii</code>                           | 585                                                                                                                                                                         |
| Environments provide by <code>enumext</code> :   |                                                                                                                                                                             |
| <code>anskey*</code>                             | 30, 69, 75, 78–81, 83, 84, 86, 87, 101, 120, 129, 131, 136, 138                                                                                                             |
| <code>enumext*</code>                            | 27, 28, 31–33, 37–41, 43, 44, 46–50, 52, 59, 60, 63–66, 68–71, 73–83, 86, 88, 89, 94, 95, 100, 101, 106, 107, 114, 115, 117, 118, 120, 122–126, 128–133, 137, 140, 141      |
| <code>enumext</code>                             | 27, 28, 32, 33, 37–41, 43–48, 50–55, 57, 60–62, 64–66, 68–71, 73–79, 81–83, 86, 88, 89, 92–95, 98, 101, 104, 105, 109, 114, 116, 119, 120, 122, 125, 130–133, 137, 138, 140 |

|                                   |                                                                                                                                                                                                 |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>keyans*</code>              | 27, 28, 30–34, 37–40, 43–50, 52, 59, 60, 63, 69, 70, 73, 74, 76, 84, 88, 94, 100, 106, 107, 114, 116, 124, 125, 137, 139, 141                                                                   |
| <code>keyanspic</code>            | 27, 28, 30, 31, 34, 40, 45, 69, 70, 73, 76, 84, 88–90, 94, 106–111, 113, 139                                                                                                                    |
| <code>keyans</code>               | 27, 28, 30, 31, 33, 34, 37, 38, 40, 41, 45–48, 50, 52, 54, 57, 60–63, 69, 70, 73, 74, 76, 84, 88–90, 94, 96–98, 104–106, 108–110, 112, 116, 126, 137, 139                                       |
| Environments:                     |                                                                                                                                                                                                 |
| <code>center</code>               | 114                                                                                                                                                                                             |
| <code>description</code>          | 94, 114                                                                                                                                                                                         |
| <code>enumerate</code>            | 114                                                                                                                                                                                             |
| <code>flushleft</code>            | 114                                                                                                                                                                                             |
| <code>flushright</code>           | 114                                                                                                                                                                                             |
| <code>itemize</code>              | 114                                                                                                                                                                                             |
| <code>list</code>                 | 32, 35, 36, 81, 94, 98, 102, 104, 106, 108–110, 114, 117                                                                                                                                        |
| <code>lrbox</code>                | 123                                                                                                                                                                                             |
| <code>minipage</code>             | 32, 35, 36, 38, 39, 52, 54–56, 108, 109, 111, 113, 114, 117, 123                                                                                                                                |
| <code>multicols</code>            | 52–56, 60, 102, 103                                                                                                                                                                             |
| <code>quotation</code>            | 114                                                                                                                                                                                             |
| <code>quote</code>                | 114                                                                                                                                                                                             |
| <code>scontents</code>            | 84, 86                                                                                                                                                                                          |
| <code>tabbing</code>              | 114                                                                                                                                                                                             |
| <code>trivlist</code>             | 114                                                                                                                                                                                             |
| <code>verbatim</code>             | 114                                                                                                                                                                                             |
| <code>verse</code>                | 114                                                                                                                                                                                             |
| exp commands:                     |                                                                                                                                                                                                 |
| <code>\exp_after:wN</code>        | 5175                                                                                                                                                                                            |
| <code>\exp_args:Ne</code>         | 2990, 2998, 3665, 5163                                                                                                                                                                          |
| <code>\exp_args:NV</code>         | 2710, 2865, 3381, 3399, 3421, 5448                                                                                                                                                              |
| <code>\exp_not:N</code>           | 54, 625, 752, 796, 812, 864, 1060, 1063, 1074, 1075, 1076, 1087, 1088, 1099, 1100, 2649, 2681, 2682, 3141, 3206, 3207, 3219, 3220, 5046, 5047, 5172                                             |
| <code>\exp_not:n</code>           | 287, 302, 315, 323, 331, 691, 711, 752, 796, 812, 864, 1061, 1854, 1863, 2316, 2413, 2425, 2587, 2615, 2625, 2635, 2649, 2650, 2957, 2970, 2980, 3105, 3143, 3145, 4077, 5277, 5287, 5480, 5485 |
| F                                 |                                                                                                                                                                                                 |
| <code>\fbox</code>                | 2297                                                                                                                                                                                            |
| <code>\fboxrule</code>            | 2297                                                                                                                                                                                            |
| <code>\fboxsep</code>             | 2297                                                                                                                                                                                            |
| file commands:                    |                                                                                                                                                                                                 |
| <code>\file_input_stop:</code>    | 5884                                                                                                                                                                                            |
| <code>first</code>                | 1113                                                                                                                                                                                            |
| <code>font</code>                 | 644                                                                                                                                                                                             |
| <code>\footnote</code>            | 37                                                                                                                                                                                              |
| <code>\footnote</code>            | 37, 442, 472                                                                                                                                                                                    |
| <code>\footnotemark</code>        | 452, 482                                                                                                                                                                                        |
| <code>\footnotesize</code>        | 2682, 3207, 3220, 5047                                                                                                                                                                          |
| <code>\footnotetext</code>        | 438                                                                                                                                                                                             |
| <code>\foreachkeyans</code>       | 17, 134, 5413                                                                                                                                                                                   |
| G                                 |                                                                                                                                                                                                 |
| <code>\getkeyans</code>           | 17, 129, 5161                                                                                                                                                                                   |
| group commands:                   |                                                                                                                                                                                                 |
| <code>\group_begin:</code>        | 2680, 2725, 2900, 2987, 3205, 3218, 5045, 5218                                                                                                                                                  |
| <code>\group_end:</code>          | 2687, 2741, 3004, 3212, 3225, 5052, 5225                                                                                                                                                        |
| H                                 |                                                                                                                                                                                                 |
| <code>\hbadness</code>            | 4856, 5132                                                                                                                                                                                      |
| hbox commands:                    |                                                                                                                                                                                                 |
| <code>\hbox_overlap_left:n</code> | 3286, 4814                                                                                                                                                                                      |
| <code>\hbox_set:Nn</code>         | 614, 4123                                                                                                                                                                                       |

`\hbox_set_end:` ..... 4855, 5131  
`\hbox_set_to_wd:Nnw` ..... 4832, 5104  
`\hfill` 674, 679, 685, 686, 1672, 1699, 2649, 3141, 4528, 4592  
hook commands:  
`\hook_gput_code:nnn` ..... 5, 205, 209, 213, 386  
`\hook_gremove_code:nn` ..... 86, 2916  
`\hook_gset_rule:nnnn` ..... 387  
`\hook_if_empty:nTF` ..... 2914  
`\hyperlink` ..... 80, 90  
`\hyperlink` ..... 2649, 3141  
`\hypertarget` ..... 36  
`\hypertarget` ..... 413

## I

`\IfDocumentMetadataTF` .. 504, 520, 534, 547, 3303, 3478,  
3992, 4000, 4008, 4044, 4052, 4060, 4146, 4155, 4163,  
4170, 4175, 4223, 4232, 4316, 4324, 4526, 4590, 4636,  
4644, 4790, 4898, 4906  
`\IfHyperBoolean` ..... 394  
`\IfPackageLoadedTF` ..... 7, 15, 390, 403  
`\ignorespaces` .. 1063, 1076, 1088, 1100, 4641, 4718, 4751,  
4774, 4781, 4827, 4847, 4903, 4961, 4995, 5027, 5099,  
5119  
`\inputlineno` ..... 287, 302, 315, 323, 331  
int commands:  
`\int_add:Nn` ..... 4436, 4485  
`\int_case:nn` ... 1237, 1362, 2162, 2188, 2227, 2251  
`\int_case:nnTF` ..... 241  
`\int_compare:nNnTF` .. 568, 784, 800, 819, 826, 1332,  
1351, 1505, 1523, 1635, 1654, 1666, 1694, 2275, 2281,  
2749, 2753, 2757, 2765, 2811, 2815, 2819, 3016, 3037,  
3078, 3083, 3088, 3113, 3201, 3647, 3658, 3680, 3693,  
3709, 3724, 3739, 3780, 3845, 3849, 3877, 3902, 3918,  
4095, 4206, 4210, 4406, 4416, 4432, 4455, 4465, 4481,  
4654, 4658, 4696, 4706, 4858, 4870, 4919, 4931, 5134,  
5146, 5331, 5463  
`\int_compare_p:nNn` ... 256, 266, 278, 279, 293, 294,  
1641, 1642, 2168, 2194, 2532, 2542, 2554, 2555, 2570,  
2611, 2788, 2789, 2800, 2801, 2953, 3498, 3690, 5063  
`\int_decr:N` ..... 4435, 4484  
`\int_eval:n` .. 373, 918, 2431, 2582, 2682, 3096, 3207,  
3220, 3581, 3626, 4424, 4473, 5047  
`\int_from_alph:n` ..... 883, 897  
`\int_from_roman:n` ..... 885, 899  
`\int_gadd:Nn` ..... 4437, 4486  
`\int_gdecr:N` ..... 2171, 2176, 2180, 2184, 2197  
`\int_gincr:N` 2004, 2009, 2594, 3151, 3240, 3274, 3456,  
3769, 3869, 4247, 4724, 4800, 4967, 5036  
`\int_gset:Nn` ..... 450, 480, 2220  
`\int_gset_eq:NN` .. 447, 477, 1903, 1910, 1916, 1922,  
1930, 1937, 1943, 1949  
`\int_gzero:N` . 344, 345, 346, 1680, 1707, 2287, 3009,  
3785, 3923, 4881, 5158  
`\int_if_exist:NnTF` 1878, 1914, 1920, 1941, 1947, 2125  
`\int_incr:N` 2764, 3464, 3646, 3840, 4094, 4653, 4723,  
4918, 4966, 5001  
`\int_mod:nn` ..... 4872, 5148  
`\int_new:N` . 24, 25, 26, 27, 28, 29, 57, 58, 83, 100, 119,  
140, 141, 152, 154, 155, 156, 158, 169, 170, 178, 179,  
180, 181, 182, 1880, 2128  
`\int_set:Nn` 879, 883, 885, 2017, 2024, 2036, 2045, 2901,  
4310, 4311, 4351, 4382, 4405, 4411, 4427, 4454, 4460,  
4476, 4856, 5132, 5327, 5465  
`\int_set_eq:NN` ..... 2005, 2010, 4434, 4483  
`\int_sign:n` ..... 2222  
`\int_step_function:nnN` ..... 2548, 2561, 2575

`\int_step_function:nnnN` ..... 5469  
`\int_step_inline:nn` ..... 5379  
`\int_step_inline:nnn` ..... 4312  
`\int_to_roman:n` ..... 217, 2528, 2565  
`\int_use:N` 366, 371, 372, 1333, 1352, 1667, 2019, 2026,  
2038, 2047, 3581, 3601, 3626, 3666, 3710, 3719, 3734,  
3740, 4409, 4410, 4422, 4458, 4459, 4471, 5800, 5804,  
5810, 5814  
`\int_zero:N` ..... 3471, 4862, 4986, 5138  
`\item` . 91, 96, 120, 123, 126, 128, 379, 2449, 2455, 2480, 2486,  
2608, 3115, 3118, 3292, 3460, 4150, 4151, 4638, 4640,  
4900, 4902, 5034  
`\item*` ..... 5, 15, 73, 3458  
`item-pos*` ..... 3355  
`item-sym*` ..... 3355  
`\itemindent` ..... 99  
`\itemindent` ..... 98  
`itemindent` ..... 1007  
`\itemsep` ..... 4139  
`\itemwidth` . 584, 2297, 3800, 3806, 3937, 3943, 4445, 4449,  
4494, 4498

## K

`keyans` ..... 14, 3946  
`keyans*` ..... 14, 4888  
`keyanspic` ..... 15, 4141

Keys for `\anskey` provide by [enumext](#):

`break-col` ..... 79, 81, 84–86  
`item-join` ..... 80, 81, 84–86  
`item-pos*` ..... 80, 81, 84–86  
`item-star` ..... 80, 81, 84–86  
`item-sym*` ..... 80, 81, 84–86

Keys for `anskey*` provide by [enumext](#):

`break-col` ..... 79, 81, 84–86  
`item-join` ..... 80, 81, 84–86  
`item-pos*` ..... 80, 81, 84–86  
`item-star` ..... 80, 81, 84–86  
`item-sym*` ..... 80, 81, 84–86

Keys for environments provide by [enumext](#):

`above*` ..... 29, 48, 61–63, 102, 120  
`above` ..... 29, 48, 61–63, 102, 120, 125  
`after` ..... 50, 51, 103, 120, 125  
`align` ..... 29, 42, 91, 93, 97, 122, 136  
`base-fix` ..... 47, 48, 64, 76, 100  
`before*` ..... 50, 102, 120, 125  
`before` ..... 50  
`below*` ..... 29, 61–63, 103, 120  
`below` ..... 29, 61–63, 103, 120, 125  
`check-ans` . 31, 32, 34, 68–70, 72, 73, 76, 87, 90, 103, 104,  
120, 124, 138  
`columns-sep` ..... 52, 102, 124  
`columns` ..... 29, 52, 61, 102  
`first` ..... 50, 51, 123  
`font` ..... 41, 93, 97, 112, 122  
`item-pos*` ..... 92, 94  
`item-sym*` ..... 30, 92, 94  
`itemindent` ..... 29, 48, 49, 92, 96, 123  
`itemsep` ..... 47, 100, 124  
`label-pos` ..... 108, 110–112  
`label-sep` ..... 109  
`labelsep` ..... 41, 98, 122  
`labelwidth` ..... 40, 41, 43, 45, 46, 98, 122  
`label` ..... 28, 40–42, 45, 46, 109, 114  
`layout-sep` ..... 109  
`layout-sty` ..... 109, 113

©2024 by Pablo González L



`\msg_fatal:nnn` ..... 588  
`\msg_info:nnn` ..... 9, 12, 17, 20, 392, 405  
`\msg_line_context:` .. 5521, 5526, 5531, 5560, 5565, 5570, 5585, 5600, 5604, 5608, 5612, 5616, 5620, 5627, 5634, 5640, 5654, 5658, 5663, 5667, 5671, 5675, 5680, 5684, 5688, 5692, 5697, 5732, 5736, 5741, 5746, 5750, 5755, 5831, 5835, 5840, 5845, 5850, 5854, 5858, 5862, 5866, 5870, 5874, 5878, 5882  
`\msg_log:nnn` ..... 2117, 2122, 2127  
`\msg_log:nnnnn` ..... 370, 2260, 2265, 2270  
`\msg_log:nnnnnn` ..... 362  
`\msg_new:nnn` 5488, 5492, 5496, 5500, 5505, 5518, 5523, 5528, 5533, 5542, 5550, 5554, 5558, 5563, 5568, 5583, 5598, 5602, 5606, 5610, 5614, 5618, 5622, 5631, 5637, 5643, 5647, 5651, 5656, 5661, 5665, 5669, 5673, 5678, 5682, 5686, 5690, 5695, 5730, 5734, 5739, 5744, 5748, 5753, 5829, 5833, 5838, 5843, 5848, 5852, 5856, 5860, 5864, 5868, 5872, 5876, 5880  
`\msg_new:nnnn` .. 5509, 5700, 5709, 5718, 5724, 5757, 5767, 5777, 5787, 5797, 5807, 5817, 5823  
`\msg_term:nnnn` . 2081, 2086, 3590, 3600, 3632, 3637  
`\msg_term:nnnnn` ..... 2241  
`\msg_warning:nn` ..... 3782, 3920  
`\msg_warning:nnnn` 2278, 2284, 3538, 3543, 4408, 4421, 4457, 4470  
`\msg_warning:nnnnn` ..... 2236, 2246  
`\multicolsep` ..... 102  
`\multicolsep` ..... 1336, 1508, 3730, 3893

N

`\NeedsTeXFormat` ..... 3  
`\NewCommandCopy` ..... 379  
`\newcounter` ..... 591  
`\NewDocumentCommand` 1633, 2722, 4200, 5161, 5216, 5323, 5372, 5450  
`\NewDocumentEnvironment` . 3812, 3946, 4141, 4625, 4888  
`\newenvsc` ..... 2836  
`\newlabel` ..... 37  
`\newlabel` ..... 425  
`no-store` ..... 2131  
`\noindent` ..... 3771, 4516, 4581, 4861, 5137  
`\nointerlineskip` 1345, 1348, 1517, 1520, 1674, 1701, 4516, 4581  
`noitemsep` ..... 931  
`\nopagebreak` 1282, 1310, 1345, 1348, 1517, 1520, 1624, 1630  
`\normalfont` ..... 2681, 3206, 3219, 5046  
`nosep` ..... 931

P

Packages:  
`caption` ..... 117  
`enumext` ..... 27, 40, 42, 68, 94, 98, 108, 136  
`enumitem` ..... 40  
`expl3` ..... 114  
`footnotehyper` ..... 36, 38, 39  
`hyperref` ..... 31, 32, 36, 37, 80, 90, 122, 136  
`latex-lab-block` ..... 36  
`ltxcmd` ..... 36  
`ltsockets` ..... 106  
`lua-visual-debug` ..... 55  
`multicol` ..... 27, 136  
`scontents` ..... 27, 83, 84  
`shortlst` ..... 114, 118, 123  
`tagpdf` ..... 106

`\par` .. 1282, 1310, 1348, 1520, 1624, 1630, 1669, 1674, 1696, 1701, 2657, 3747, 3908, 3926, 4186, 4189, 4329, 4543, 4558, 4604, 4618, 4861, 5137  
para commands:  
`\para_end:` ..... 4878, 5155  
`\parbox` ..... 2297  
`\parindent` ..... 4843, 5115  
`\parsep` ..... 53, 109  
`\parsep` ..... 991, 3623, 4119, 4128  
`parsep` ..... 931  
`\parskip` ..... 4844, 5116  
`\partopsep` ..... 3624, 3924, 4138  
`partopsep` ..... 931  
peek commands:  
`\peek_meaning:NTF` 4729, 4743, 4758, 4769, 4972, 4987, 5003  
`\peek_meaning_remove:NTF` ..... 4736, 4979  
`\peek_remove_spaces:n` ..... 3465  
`\phantomsection` ..... 36  
`\phantomsection` ..... 414  
prg commands:  
`\prg_do_nothing:` ..... 418  
`\prg_new_protected_conditional:Npnn` ... 219  
`\prg_replicate:nn` ..... 236  
`\prg_return_false:` ..... 223  
`\prg_return_true:` ..... 222  
`\printkeyans` ..... 18, 130, 5216  
prop commands:  
`\prop_const_from_keyval:Nn` ..... 5364  
`\prop_count:N` 364, 2431, 2582, 2684, 3096, 3209, 3222, 5049, 5466  
`\prop_get:NnNTF` ..... 5390  
`\prop_gput_if_not_in:Nnn` ..... 2429  
`\prop_if_exist:NTF` ..... 2115, 5181, 5459  
`\prop_item:Nn` ..... 5183, 5483  
`\prop_new:N` ..... 2118  
`\ProvidesExplPackage` ..... 4

R

`\raggedcolumns` ..... 3733, 3896  
`\raisebox` ..... 4271  
`\ref` ..... 78, 88  
`ref` ..... 717, 763, 834  
`\refstepcounter` ..... 4797, 5082  
regex commands:  
`\regex_match:nnTF` .. 221, 882, 884, 896, 898, 2929  
`\regex_replace_once:nnN` ..... 229  
`\renewcommand` ..... 752, 796, 812, 864  
`\RenewDocumentCommand` . 442, 472, 1684, 1711, 3292, 3319, 3335, 3460, 3512, 3522, 4151  
`\RequirePackage` ..... 13, 21  
`resume` ..... 1827  
`resume*` ..... 1827  
`rightmargin` ..... 1007  
`\Roman` ..... 41, 45, 46  
`\Roman` ..... 610  
`\roman` ..... 41, 45, 46  
`\roman` ..... 611, 735, 5206

S

`\s` ..... 2930  
`save-ans` ..... 2070  
`save-key` ..... 2370  
`save-ref` ..... 2290  
`save-sep` ..... 2290

## scan commands:

\scan\_stop: . . . . . 4150, 4638, 4900, 5172, 5175

## scontents internal commands:

\l\_scontents\_fname\_out\_tl . . . . . 2889  
 \\_scontents\_parse\_environment\_keys:n . 2895  
 \\_scontents\_rescan\_tokens:n . . . . . 2902  
 \l\_scontents\_storing\_bool . . . . . 2887  
 \l\_scontents\_writing\_bool . . . . . 2888

## seq commands:

\seq\_clear:N . . . . . 5325, 5468  
 \seq\_const\_from\_clist:Nn . . . . . 5318  
 \seq\_count:N . . . . . 365, 4335, 5329  
 \seq\_gclear:N . . . . . 467, 468, 497, 498  
 \seq\_gput\_right:Nn . . . . . 453, 454, 483, 484, 2438  
 \seq\_if\_empty:NTF . . . . . 460, 490, 5231, 5343  
 \seq\_if\_exist:NTF . . . . . 2120, 5229  
 \seq\_if\_in:NnTF . . . . . 5235  
 \seq\_item:Nn . . . . . 2927, 4322  
 \seq\_map\_function:NN . . . . . 5334  
 \seq\_map\_inline:Nn . . . . . 5244, 5252, 5344, 5345  
 \seq\_map\_pairwise\_function:NNN . . . . 462, 492  
 \seq\_new:N 120, 121, 123, 138, 171, 172, 173, 174, 2123  
 \seq\_pop\_left:NN . . . . . 5333  
 \seq\_put\_right:Nn . . . . . 4214, 5341, 5357, 5478  
 \seq\_set\_from\_clist:Nn . . . . . 5326  
 \seq\_set\_map\_e:NNn . . . . . 5335  
 \seq\_use:Nn . . . . . 201, 202, 5474

## series . . . . . 1827

\setcounter . . . . . 893, 897, 899, 3581, 3626, 4183  
 \setenumext . . . . . 6, 132, 5323  
 \setenumextmeta . . . . . 6, 133, 5364  
 show-ans . . . . . 2290, 2334  
 show-length . . . . . 1104  
 show-pos . . . . . 2334

## skip commands:

\skip\_add:Nn 1242, 1251, 1260, 1273, 1277, 1301, 1305,  
 1321, 1379, 1381, 1395, 1398, 1419, 1421, 1435, 1438,  
 1458, 1460, 1474, 1477, 1496, 1545, 1546, 1557, 1559,  
 4128, 4136  
 \skip\_gset:Nn . . . . . 1572, 1576, 1580  
 \skip\_gzero\_new:N . . . . . 1567, 1568  
 \skip\_horizontal:N . . 1075, 1087, 1099, 4815, 4827,  
 4865, 5099, 5141  
 \skip\_horizontal:n . . 1061, 2506, 2514, 3285, 3287,  
 4714, 4813, 4847, 4957, 5119  
 \skip\_if\_eq:nnTF 1240, 1249, 1258, 1365, 1405, 1445,  
 1533, 1569, 1591, 1735, 1749, 1763, 1774, 1785, 1796,  
 1807, 1818  
 \skip\_new:N 77, 78, 79, 84, 85, 86, 87, 88, 89, 144, 193  
 \skip\_set:Nn 1225, 1229, 1287, 1291, 1315, 1368, 1369,  
 1387, 1408, 1409, 1427, 1447, 1448, 1466, 1490, 1536,  
 1537, 1551, 1571, 1575, 1593, 1597, 1601, 1607, 1611,  
 1615, 4112  
 \skip\_set\_eq:NN 1326, 1327, 1329, 1336, 1501, 1502,  
 1503, 1508, 3579, 3622, 3623, 4844, 5116  
 \skip\_sub:Nn 1375, 1377, 1391, 1393, 1415, 1417, 1431,  
 1433, 1454, 1456, 1470, 1472, 1543, 1544, 1555, 1556  
 \skip\_use:N 1227, 1231, 1275, 1279, 1283, 1303, 1307,  
 1317, 1323, 1736, 1740, 1743, 1750, 1754, 1757, 3747  
 \skip\_vertical:N . 575, 578, 999, 4542, 4556, 4880,  
 5157  
 \skip\_vertical:n . . . . . 998, 4879, 5156  
 \skip\_zero:N 1335, 1349, 1487, 1488, 1489, 1507, 1521,  
 3624, 3730, 3893, 4138, 4139  
 \skip\_zero\_new:N . . . . . 1566, 1588, 1589, 1590

\c\_zero\_skip . 575, 578, 999, 1240, 1249, 1258, 1406,  
 1445, 1569, 1591, 1736, 1750, 1763, 1774, 1785, 1796,  
 1807, 1818, 4542, 4556, 4880, 5157

\small . . . . . 5194, 5198, 5202, 5206, 5210, 5214  
 \smash . . . . . 3337, 3524

## socket commands:

\socket\_assign\_plug:nn . . 3994, 4002, 4010, 4046,  
 4054, 4062  
 \socket\_new:nn . . . . . 3964, 4014  
 \socket\_new\_plug:nnn 3965, 3973, 3981, 4015, 4023,  
 4032  
 \socket\_use:n . . . . . 4047, 4055, 4063  
 \socket\_use:nn . . . . . 3995, 4003, 4011

start . . . . . 905

start\* . . . . . 905

start-list-tags . . . . . 3964, 4014

\stepcounter . . . . . 446, 476, 4122, 4264

stop-list-tags . . . . . 3964, 4014

stop-start-tags . . . . . 3964, 4014

## str commands:

\c\_backslash\_str 2775, 5521, 5526, 5531, 5536, 5538,  
 5540, 5545, 5547, 5645, 5649, 5653, 5663, 5667, 5675,  
 5676, 5680, 5692, 5693, 5697, 5698, 5719, 5721, 5725,  
 5727, 5755, 5818, 5820, 5824, 5826, 5835, 5836, 5840,  
 5845, 5846, 5850, 5854, 5858  
 \c\_colon\_str . . . . . 2581, 3095, 5172  
 \c\_left\_brace\_str . . . . . 5626, 5633, 5639  
 \c\_right\_brace\_str . . . . . 5626, 5633, 5639  
 \str\_case:nn . . . . . 249, 308  
 \str\_case:nnTF . 1850, 1858, 2409, 2417, 5272, 5281  
 \str\_clear:N . . . . . 3657, 4670  
 \str\_count:n . . . . . 236  
 \str\_if\_empty:NTF . . . . . 1867, 1908, 1935  
 \str\_if\_eq:nnTF . . . . . 3582, 3628, 5374  
 \str\_if\_in:nnTF . . . . . 5168  
 \str\_new:N . . . . . 80, 128, 143, 188  
 \str\_set:Nn . 675, 681, 687, 706, 707, 708, 2312, 2313,  
 2339, 2340, 4071, 4074  
 \str\_use:N . . . . . 3341

\strut . . . . . 3337, 3524

\strutbox . 1354, 1357, 1368, 1369, 1380, 1382, 1397, 1400,  
 1408, 1409, 1420, 1422, 1437, 1440, 1447, 1448, 1459,  
 1461, 1476, 1479, 1525, 1528, 1536, 1537, 1545, 1546,  
 1558, 1560, 1571, 1572, 1575, 1582, 1595, 1603, 1609,  
 1617, 4131, 4136, 4186, 4194, 4277

## T

## tag commands:

\tag\_mc\_begin:n . . . . . 3971, 4021, 4030  
 \tag\_mc\_begin\_pop:n . . . . 3987, 4039, 4178, 4180  
 \tag\_mc\_end: . . . . . 3975, 4025, 4034  
 \tag\_mc\_end\_push: . . . . . 3968, 4018, 4166  
 \tag\_resume:n . . 3967, 4017, 4157, 4165, 4234, 4326,  
 4526, 4590  
 \tag\_struct\_begin:n . 3969, 3970, 3977, 3978, 3979,  
 4019, 4020, 4027, 4028, 4029, 4167  
 \tag\_struct\_end:n 3976, 3983, 3984, 3985, 3986, 4026,  
 4035, 4036, 4037, 4038, 4177, 4179, 4644, 4906  
 \tag\_suspend:n . 3988, 4040, 4148, 4159, 4172, 4225,  
 4318, 4636, 4898  
 \tag\_tool:n . . . . . 4158

TeX and  $\TeX$  2<sub>ε</sub> commands:

\@auxout . . . . . 423  
 \@currenvir . . . . . 249, 308  
 \protected@write . . . . . 423

tex commands:

`\tex_newlinechar:D` . . . . . 2901

text commands:

`\text_expand:n` . . . . . 5164

`\textasteriskcentered` . . . . . 2309, 3361

`\textreferencemark` . . . . . 2326

`\thepage` . . . . . 429

tl commands:

`\c_space_tl` 3177, 5570, 5585, 5608, 5612, 5799, 5800, 5809, 5810, 5870, 5874

`\tl_clear:N` . . 673, 680, 2288, 2356, 2366, 2387, 2395, 2601, 2921, 2922, 3036, 3112, 5009

`\tl_clear_new:N` . . . . . 620

`\tl_const:Nn` . . . . . 46, 604

`\tl_gclear:N` . 356, 357, 358, 1888, 1893, 3011, 3330, 3350, 4562, 4622, 4816

`\tl_gclear_new:N` . . . . . 1875

`\tl_gput_right:Nn` . . . . . 605

`\tl_greplace_all:Nnn` . . . . . 626

`\tl_gset:Nn` 284, 285, 299, 300, 1876, 1889, 1894, 2113, 2925, 3261, 4764

`\tl_gset_eq:NN` . . . . . 622, 3257, 4809

`\tl_if_blank:nTF` 2714, 2732, 2869, 3385, 3403, 3425, 4807, 5438

`\tl_if_empty:NTF` . 740, 758, 786, 802, 821, 828, 854, 870, 1901, 1906, 1928, 1933, 1991, 2055, 2063, 2092, 2152, 2445, 2476, 2621, 2966, 2988, 3018, 3046, 3122, 3171, 3282, 4333, 5012, 5355

`\tl_if_empty:nTF` . . . . . 1956

`\tl_if_exist:NTF` . . . . . 1961

`\tl_if_novalue:nTF` 444, 474, 2728, 3044, 3120, 3156, 3236, 3255, 3263, 3435, 3655, 4103, 4668, 4938, 5010

`\tl_map_inline:Nn` . . . . . 227, 623

`\tl_new:N` 38, 39, 40, 43, 48, 49, 52, 53, 59, 61, 62, 64, 65, 101, 102, 103, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 122, 124, 125, 126, 129, 132, 133, 151, 161, 162, 163, 166, 187

`\tl_put_left::Ne` . . . . . 2955

`\tl_put_left:Nn` 2453, 2484, 2606, 2949, 2962, 2968, 2978, 3188, 3228, 4546, 4607, 5031, 5034

`\tl_put_right:Nn` 621, 750, 794, 810, 862, 2457, 2488, 2535, 2545, 2558, 2573, 2579, 2584, 2608, 2613, 2620, 2623, 2633, 2638, 2641, 2647, 3039, 3042, 3048, 3053, 3080, 3085, 3090, 3093, 3102, 3115, 3118, 3124, 3129, 3139, 5014, 5018

`\tl_remove_all:Nn` . . . . . 5354

`\tl_remove_once:Nn` . . . . . 2523, 3065

`\tl_replace_all:Nnn` . . . . . 625, 5389

`\tl_reverse:N` . . . . . 2522, 2524, 3064, 3066

`\tl_set:Nn` . 54, 253, 263, 312, 313, 320, 321, 328, 329, 590, 674, 679, 685, 686, 739, 783, 853, 1058, 1072, 1085, 1097, 1990, 2091, 2357, 2367, 2388, 2396, 2678, 2889, 3158, 3203, 3216, 5020, 5043, 5352, 5388, 5458

`\tl_set_eq:NN` 631, 745, 748, 791, 793, 807, 809, 859, 861, 2521, 3063, 3076, 3447, 3452, 4252, 4254

`\tl_to_str:n` . . . . . 1961, 1967, 1972, 5164

`\tl_trim_spaces:n` . . . 621, 5341, 5352, 5358, 5374

`\tl_use:N` 627, 630, 760, 823, 830, 872, 1130, 1134, 1138, 1142, 1146, 1150, 1154, 1158, 1162, 1166, 1170, 1174, 1178, 1182, 1186, 1190, 2511, 2528, 2536, 2547, 2560, 2565, 2576, 3244, 3250, 3278, 3321, 3322, 3329, 3343, 3438, 3442, 3450, 3514, 3515, 3517, 3528, 3819, 3952, 4257, 4553, 4614, 4820, 4848, 4849, 5091, 5120, 5125, 5219, 5220, 5221, 5222, 5223, 5240, 5337, 5456

token commands:

`\token_to_str:N` . . . . . 425

`\topsep` . . . . . 3924, 4136

`topsep` . . . . . 931

`\topskip` . . . . . 1335, 1507

U

`\u` . . . . . 230, 2930

`\unkern` . . . . . 244

`unknown` . . . . . 3371, 3393, 3411

`\unskip` . . . . . 243

use commands:

`\use:N` . . . . . 237, 3326, 3347, 3821

`\use:n` . . . . . 1841, 2400, 5170, 5263

`\use_none:nn` . . . . . 417, 5395

`\usecounter` . . . . . 3580, 3625

V

`\value` . . . . 1904, 1910, 1917, 1923, 1931, 1937, 1944, 1950

vbox commands:

`\vbox_set:Nn` . . . . . 4227

`\vbox_set_top:Nn` . . . . . 4551, 4612

`\vspace` 991, 1740, 1743, 1754, 1757, 1767, 1769, 1778, 1780, 1789, 1791, 1800, 1802, 1811, 1813, 1822, 1824

W

`widest` . . . . . 905

`wrap-ans` . . . . . 2290

`wrap-key` . . . . . 2290

`wrap-label` . . . . . 644

`wrap-label*` . . . . . 644

`wrap-opt` . . . . . 2290

Z

`\z` . . . . . 2930