

marginalia — Marginal content anywhere with automatic adjustment for Lua¹TeX¹

¹ This document describes v0.83.30, last revised 2026-07-28.

² a.j.cain (AT) gmail.com

Alan J. Cain²

Released 2026-07-28

Abstract

This Lua¹TeX package allows the placement of marginal content anywhere, without `\marginpar`'s limits, and automatically adjusts positions to prevent overlaps or content being pushed off the page, and offers key-value settings that allow fine-grained customization.

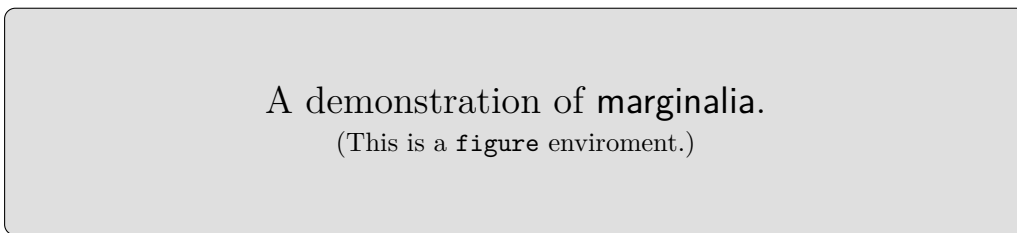
Demonstration

The `marginalia` package permits intelligent placement of marginal content, such as notes. This page demonstrates some of its capabilities.

Key-value options allow global or local settings of the style, width, placement, and spacing of marginal content. In particular, styles and widths can be specified globally depending on the margin in which the content is placed; the correct style or width will be selected for each item. For example, in this demonstration, 15 mm-wide ragged-right sans-serif has been specified globally as the default for items in the right margin and 35 mm-wide justified sans-serif for items in the left margin. Note that the vertical position of items is adjusted automatically to avoid overlaps, but there is an option to specify that items have fixed positions.

By default, this marginal note would have its top line aligned with the last line of the adjacent paragraph, but it has been automatically moved upwards to avoid clashing with the figure caption below, which has been set with an option that fixes its position.

Figure 0
`marginalia` can place content alongside floats, so it can be used to place float captions in the margin. (Here, the text style has been locally switched to Roman.)



(Like this one.)

Notes can be placed on both sides of the paragraph.

This is a top-aligned margin item pointing to the relevant line of text.

Unlike the usual `\marginpar` command, `marginalia` allows marginal content to be placed next to floats, footnotes, or the page head or footer. 'Marks' can also be added pointing to the relevant line of text and there are various vertical alignment options.

The automatic adjustment of the vertical positions of items will not result in items being closer than specified distances from the top or bottom of the page (unless there is actually insufficient space to place the items).

This marginal note, the width and style of which have locally been set to 25 mm and ragged left, would by default have its top line aligned with the last line of the adjacent paragraph, but it has been moved upwards to maintain a minimum distance of 10 mm from the bottom of the page.

This is a middle-aligned margin item pointing to the relevant line of text.

Contents

Demonstration	1
1 Introduction	3
2 Requirements	4
3 Installation	5
4 Getting started	5
5 User commands	5
5.1 Access to page and column	5
6 Options	7
6.1 Type	7
6.2 Horizontal placement	7
6.3 Vertical placement	9
6.4 Appearance	11
6.5 Warning message configuration	14
7 Placement	14
7.1 Horizontal placement	14
7.2 Vertical placement	16
8 Usage notes	17
9 Incompatibilities	18
10 Limitations	18
11 Implementation (L^AT_EX package)	19
11.1 Initial set-up	19
11.2 Tagging set-up	19
11.3 Auxiliary macro for dimension setting	19
11.4 Auxiliary macros for setting options	20
11.5 Options	20
11.5.1 Type	21
11.5.2 Horizontal placement	21
11.5.3 Vertical placement	22
11.5.4 Appearance	24
11.5.5 Warning message configuration	26
11.6 Lua backend and interface	27
11.7 Processing data from the .aux file	28
11.8 Writing version data to the .aux file	29
11.9 Writing page data to the .aux file	29
11.10 Marginal content item processing	30
11.10.1 Variables	30
Variables set by L ^A T _E X.	30
Variables set by Lua.	31

11.10.2	Core macro	32
11.10.3	Horizontal separation, width, style, mark selection	35
11.10.4	Auxiliary box macros	36
11.10.5	Placement macros	37
11.11	User commands	38
12	Implementation (Lua backend)	38
12.1	Initial set-up	38
12.2	Message functions	39
12.3	Global variables	39
12.4	Constants	39
12.5	Keys for tables	39
12.5.1	Keys for both page and item data tables	40
12.5.2	Keys for page data tables, layout etc.	40
12.5.3	Keys for item data tables	40
12.6	Utility functions	41
12.7	Generic page/item data functions	42
12.8	Processing of page data from <code>.aux</code> file	44
12.9	Processing of item data from <code>.aux</code> file	45
12.10	Writing reports	47
12.11	Computing horizontal positions	49
12.12	Computing vertical positions	53
12.12.1	Auxiliary functions	53
12.12.2	Computing <code>optfixed</code> enabled	54
12.12.3	Computing vertical adjustment	55
12.12.4	Checking vertical adjustment	56
12.12.5	Core vertical position computation	58
12.13	Passing <code>item_data</code> back to <code>L^AT_EX</code>	60
12.14	Export public functions	61

Index 62

1 Introduction

The `LATEX` `\marginpar` command is the basic method for placing content in the margin. For purposes such as drawing attention to particular points in the text, it functions well. Its main limitation is that `\marginpar` works via the `LATEX` float mechanism and so cannot be used to create marginal content next to a figure, table, or other float, or next to a footnote, or to place running heads in the margin, such as are found in the left-hand margin of this document except for the ‘implementation’ section. (Bringhurst called this style ‘running shoulderheads’ [Bri04, p. 65], but the term may be non-standard.)

Trying to set many separate pieces of marginal content using `\marginpar` can lead to other problems. If two `\marginpars` would clash, `LATEX` shifts the second item downward. But the cumulative effect can lead to `\marginpars` being shifted downward off the bottom of the page. Further, the asynchronous nature of `TEX`’s page-breaking can cause: (1) a `\marginpar` to be placed in the wrong margin; (2) the topmost `\marginpar` on a page to be unnecessarily shifted downward because of a hypothetical clash that would have occurred with the previous `\marginpar`, had they been on the same page.

Requirements

3 URL: <https://ctan.org/pkg/mparhack>

4 URL: <https://ctan.org/pkg/marginnote>

5 URL: <https://ctan.org/pkg/marginfix>

6 URL: <https://ctan.org/pkg/marginfit>

Packages like `mparhack`³ (Tom Sgouros & Stefan Ulrich), `marginnote`⁴ (Markus Kohm), `marginfix`⁵ (Stephen Hicks) and `marginfit`⁶ (Maurice Leclaire) were created to avoid these limitations and problems. `mparhack` only ensures that each `marginpar` appears on the correct side of the page. `marginnote` allows marginal content to be placed anywhere, but does not adjust positions to avoid clashes. `marginfix` adjusts positions, but the unadjusted vertical positioning can be slightly off, and the package still uses floats. `marginfit` gets positions exactly right, but uses the insert mechanism and so marginal content cannot appear next to floats or footnotes.

This Lua $\text{\LaTeX}$ package, `marginalia`, provides a `\marginalia` command that attempts to avoid these limitations. Marginal content is placed, not via floats or inserts, but by a calculated per-item horizontal shift inside an (invisible) `\rlap` or `\llap` from the position where the `\marginalia` command was issued (which is similar to the technique used by `marginnote`), plus a calculated per-item vertical shift to avoid clashes with other content. The vertical shift is usually downward, but may be upward when necessary to prevent content from being shifted off the bottom of the page (which is similar to the vertical shifts performed by `marginfix` and `marginfit`).

The calculation of the horizontal and vertical shifts uses information written to the `.aux` file during the previous Lua $\text{\LaTeX}$ run. It thus takes at least two runs for all content to appear in the correct places. The package reports any changes from the previous run and any problems encountered.

Note: `marginalia` was written to typeset running heads in the margin, sidenote references, side-captions for floats, and small marginal figures in the author's book *Form & Number: A History of Mathematical Beauty* [Cai24].⁷ Thus the basic functionality has been tested extensively, and it has performed correctly.

7 *Form & Number* is freely available on the Internet Archive under a Creative Commons licence.

URL: https://archive.org/details/cain_formand_number_ebook_large

8 URL: <https://www.latex-project.org/lppl.txt>

Licence. `marginalia` is released under the $\text{\LaTeX}$ Project Public Licence v1.3c or later.⁸

Acknowledgements. The author thanks Ulrike Fischer for explaining how to add tagging support, and Julien Labbé and François Pantigny for some valuable suggestions.

Translation. A French translation of the documentation has been made by members of GUTenberg (Le Groupe francophone des Utilisateurs de $\text{\TeX}$).⁹

Feature requests and bug reports The development code and issue tracker are hosted at Codeberg.¹⁰

9 URL: <https://gitlab.gutenberg-asso.fr/gutenberg/traduction-de-marginalia/>

10 URL: <https://codeberg.org/ajcain/marginalia>

Other resources. An introduction to `marginalia` has appeared in *TUGboat* [Cai25].

2 Requirements

`marginalia` requires

- (1) Lua $\text{\LaTeX}$,
- (2) a recent $\text{\LaTeX}$ kernel with `expl3` support (any kernel version since 2020-02-02 should suffice).

It does not depend on any other packages.

3 Installation

To install `marginalia` manually, run `luatex marginalia.ins` and copy `marginalia.sty` and `marginalia.lua` to somewhere `Lua $\TeX$` can find them.

4 Getting started

`marginalia` works ‘out of the box’. Load the package (there are no package options) and use the main `\marginalia` command to place marginal content. Figure 4.1 shows the source code for a small demonstration and the resulting document. *The source code must be processed twice by `Lua $\TeX$` for the marginal content to be placed correctly.* (See Section 8 for discussion of the need for multiple runs.)

Turn to Section 5 for a detailed description of the available user commands, and Section 6 for the various options (such as `style=<code>`) that can be used to change the placement and formatting of the marginal content.

5 User commands

`\marginalia` `\marginalia[<options>]{<content>}`

This is the basic command for placing marginal content. The `<content>` can, roughly speaking, be anything: text, mathematics, included graphics, `TikZ`. The optional argument `<options>` is a key–value list that specifies how the content is typeset. The keys are described in Section 6.

`\marginaliasetup` `\marginaliasetup{<options>}`

This command is used to set options for subsequent calls to `\marginalia`. The argument `<options>` is the same kind of key–value list as the `<options>` argument for the `\marginalia` command, and the keys are described in Subsection 6.

Note that `\marginaliasetup` can be used in the preamble or in the body of the document. Options set using `\marginaliasetup` have effect only within the current group.

`\marginalianewgeometry` `\marginalianewgeometry`

This command signals to `marginalia` that the page layout has been changed, for instance by using the `\newgeometry` command from the `geometry` package,¹¹ or by using the $\TeX$ command `\twocolumn` to switch to two-column mode. It should be issued immediately after such a change, and certainly before the first page with the new layout has been shipped out. There is no harm in using it unnecessarily.

¹¹ URL: <https://ctan.org/pkg/geometry>

5.1 Access to page and column

Within the `<content>` of `\marginalia`, two counters are available which specify the actual page and column in which the call to `\marginalia` appears. These counters can be used to select different actions depending on the page on which the content appears or (in two-column mode) whether it pertains to the left or right column. It is best to use the variants of the `style` and `width` keys if marginal content should have different

User commands

```
\documentclass[11pt,a4paper]{article}

\usepackage{marginalia}

\begin{document}

Here is some body text.\marginalia{Here is a marginal note.} Some more
body text.\marginalia[style=\footnotesize\itshape\raggedright]{Here is another
    marginal note, set in smaller text and italics, whose position has been been
    adjusted automatically.}

\vspace{20mm}

Some final body text after a space.\marginalia[pos=left, valign=b,
style=\sffamily\raggedleft, width=35mm]{This note is placed on the left side
    of the page, wider, in sans serif, ragged left, and bottom-aligned.}

\end{document}
```

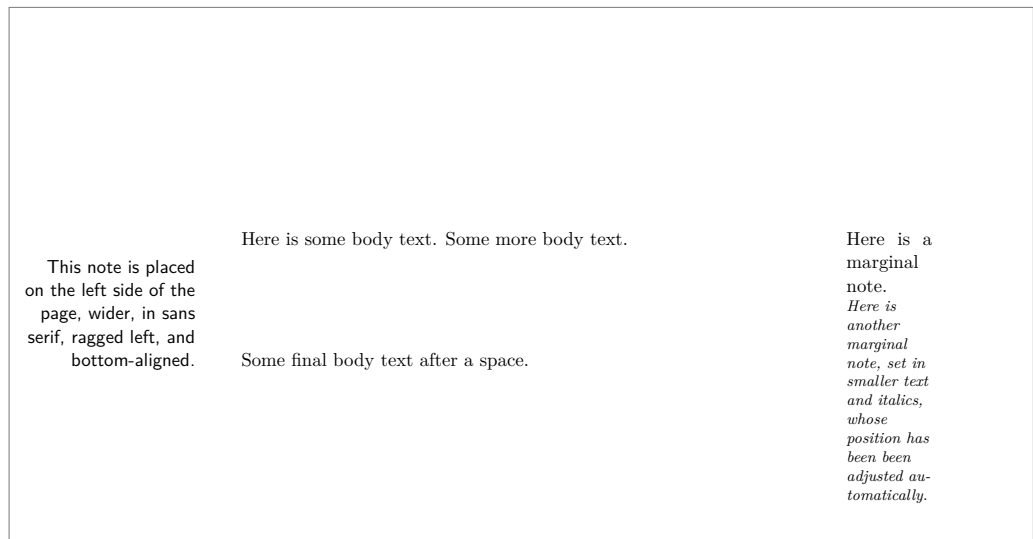


Figure 4.1: A small demonstration of `marginalia`.

widths or styles depending on whether they appear on a recto/verso page or pertain to a particular column. These counters are made available for purposes not covered by the `style` and `width` variants. The value of each counter is based on the position of the call to `\marginalia` on the previous Lua^AT_EX run.

`\marginaliapage` A counter register, available within the `<content>` of `\marginalia`, that holds the actual page on which the marginal content appears. The value is based on the previous Lua^AT_EX run and will default to 1.

<code>\marginaliacolumn</code>	A counter register, available within the <code><content></code> of <code>\marginalia</code> , that holds the actual column to which the marginal content pertains. The value is 1 for the left column, 2 for the right column. In one-column mode, the value is always 0. (If the key <code>column</code> is used to manually specify the column to which the content pertains, the value of <code>\marginaliacolumn</code> will change accordingly.) The value is based on the previous Lua ^A TeX run and will default to 0.
--------------------------------	--

6 Options

The description of keys in this section, which are summarized in [Tables 1](#) and [2](#) (on pages [8](#) and [13](#) respectively), should be read in conjunction with the discussion of how marginal content is placed in [Section 7](#). In particular, the variants of the keys `style`, `width`, and `mark` follow the terminology shown in [Figure 7.1](#).

Note that the initial values of the various keys that take dimensions (namely `width...`, `xsep...`, `ysep...`) are assigned at `\begin{document}`. Thus their defaults are determined by the values of `\marginparwidth`, `\marginparsep`, `\marginparpush` and the page layout at `\begin{document}`, *not* at package load time. Similarly, if the user sets these keys in the preamble, the assignment is evaluated at `\begin{document}`. For example, using `\marginaliasetup{width=.5\oddsidemargin}` in the preamble results in the `width` key being set to half the value of `\oddsidemargin` at `\begin{document}`, *not* to half the value of `\oddsidemargin` at the call to `\marginaliasetup`.

6.1 Type

- type** The `type` of an item of marginal content can be set to one of the following three values:
- normal:** The vertical position of the item will be changed automatically if necessary to prevent a clash with another item of content.
 - fixed:** The vertical position of the item will *never* be changed automatically from the position specified by `yshift`, even if there is a clash with another item. (The type `fixed` was designed for setting float captions in the margin, since a caption should not move away from the float with which it is associated.)
 - optfixed:** The vertical position of the item will *never* be changed automatically from the position specified by `yshift`, even if there is a clash with another item. But an `optfixed` item will not appear in the document if it would clash with a `fixed` item. (The type `optfixed` was designed for setting running heads in the margin, which should not appear if they would clash with a figure caption set in the margin.)
- (*Default: normal*)

6.2 Horizontal placement

- pos** The position in which an item of marginal content should be placed. It can be set to one of the following four values:
- auto:** Place the item in the default position as described in [Section 7](#): the outer margin in single-column mode, and on the opposite side from the other column in two-column mode.
 - reverse:** Place the item on the opposite side of the text block (in one-column mode) or column (in two-column mode) from `auto`.
 - left:** The left side of the text block or column.

Options

Table 1: Summary of keys affecting the type and position of the marginal content. (For keys affecting the appearance of marginal content, see Table 2.) All keys can be set using `\marginaliasetup` or passed in the optional argument to `\marginalia`.

Key name	Value	Default
<code>type</code>	<code>{normal, fixed, optfixed}</code>	<code>normal</code>
<code>pos</code>	<code>{auto, reverse, left, right, nearest}</code>	<code>auto</code>
<code>column</code>	<code>{auto, one, left, right}</code>	<code>auto</code>
<code>xsep</code>	Dimension	<code>\marginparsep</code>
<code>xsep outer</code>	Dimension	<code>\marginparsep</code>
<code>xsep inner</code>	Dimension	<code>\marginparsep</code>
<code>xsep between</code>	Dimension	<code>\marginparsep</code>
<code>xsep recto outer</code>	Dimension	<code>\marginparsep</code>
<code>xsep recto inner</code>	Dimension	<code>\marginparsep</code>
<code>xsep verso outer</code>	Dimension	<code>\marginparsep</code>
<code>xsep verso inner</code>	Dimension	<code>\marginparsep</code>
<code>xsep right between</code>	Dimension	<code>\marginparsep</code>
<code>xsep left between</code>	Dimension	<code>\marginparsep</code>
<code>valign</code>	<code>{t, b, c, m}</code>	<code>t</code>
<code>yshift</code>	Dimension	<code>0pt</code>
<code>ysep</code>	Dimension	<code>\marginparpush</code>
<code>ysep above below</code>	Dimension	<code>\marginparpush</code>
<code>ysep above</code>	Dimension	<code>\marginparpush</code>
<code>ysep below</code>	Dimension	<code>\marginparpush</code>
<code>ysep page top</code>	Dimension	[Margin above textblock]
<code>ysep page bottom</code>	Dimension	[Margin below textblock]
<code>ysep page top margin</code>	[None]	—
<code>ysep page bottom margin</code>	[None]	—
<code>ysep page top bottom margin</code>	[None]	—

right: The right side of the text block of column.

nearest: The side of the text block or column nearest to which `\marginalia` was called.

(Default: `auto`)

column In two-column mode, `marginalia` tries to determine to which column an item of marginal content pertains using the position of the call to `\marginalia`. If the call is to the left of the mid-point between the columns, the item is assumed to pertain to the left column; otherwise, it is assumed to pertain to the right column. In certain situations, this might lead to undesired placement of the item. In particular, any call to `\marginalia` in a full-width float in two-column mode would be handled as if it were a call from one of the columns and might thus be set in the wrong place. Similarly, an overfull hbox or a piece of `\rlap`-ped text might carry a call to `\marginalia` from the left column text into the area of the page occupied by the right column.

The key `column` can be used to specify which column `marginalia` should place the item in. It can be set to one of four values:

auto: Automatically determine which column an item of marginal content is placed in.

Options

one: Treat the item as being called from one-column mode.

left: Treat the item as pertaining to the left column.

right: Treat the item as pertaining to the right column.

The value of `column` has no effect in one-column mode. (*Default:* `auto`)

<code>xsep</code>	These keys specify the horizontal separation between an item of marginal content and the text block next to which it is placed. Which separation is used will depend on where the item is typeset. The terminology is as in Figure 7.1 .
<code>xsep outer</code>	
<code>xsep inner</code>	
<code>xsep between</code>	xsep recto outer: used for an item in the outer margin of a recto page.
<code>xsep recto outer</code>	xsep recto inner: used for an item in the inner margin of a recto page.
<code>xsep recto inner</code>	xsep verso outer: used for an item in the outer margin of a verso page.
<code>xsep verso outer</code>	xsep verso inner: used for an item in the inner margin of a verso page.
<code>xsep verso inner</code>	xsep right between: used for an item set from the right column between the columns.
<code>xsep right between</code>	
<code>xsep left between</code>	xsep left between: used for an item set from the left column between the columns.
	xsep outer: a shorthand for setting the keys <code>xsep recto outer</code> and <code>xsep verso outer</code> simultaneously to the same value.
	xsep inner: a shorthand for setting the keys <code>xsep recto inner</code> and <code>xsep verso inner</code> simultaneously to the same value.
	xsep between: a shorthand for setting the keys <code>xsep right between</code> and <code>xsep left between</code> simultaneously to the same value.
	xsep: a shorthand for setting all of these keys simultaneously.

(The shorthands `xsep outer` and `xsep inner` exist because page geometry is usually symmetrical between recto and verso pages as regards outer and inner margins. The shorthand `xsep between` exists because the space between columns, if used at all for marginal content, will often be shared equally.) Each of these keys must be set to a valid dimension. (*Default:* value of `\marginparsep` at `\begin{document}`)

6.3 Vertical placement

valign This key specifies the vertical alignment of the marginal content item, before any `yshift` or automatic adjustment is applied. It can be set to one of the following three values:

- t:** The baseline of the marginal content item is the baseline of the topmost box in its contents. Thus, if no `yshift` is specified and there is no automatic adjustment, the baseline of the topmost box of the marginal content will be vertically aligned with the line where the call to `\marginalia` is located.
- b:** The baseline of the marginal content item is the baseline of the bottommost box in its contents. Thus, if no `yshift` is specified and there is no automatic adjustment, the baseline of the bottommost box of the marginal content will be vertically aligned with the line where the call to `\marginalia` is located.
- c:** The baseline of the marginal content item will be placed centrally between top and bottom of the contents as a whole.
- m:** The baseline of the marginal content item will be midway between the baselines of the topmost and bottommost boxes in its contents. Thus, if the marginal content comprises an *odd* number of *equally-spaced* lines, the baseline of the middle line will be vertically aligned with the line where the call to `\marginalia` is located.

These values are illustrated in [Figure 6.1](#) If `type=normal`, then the alignments described above for `t`, `b`, `m` may not hold because of automatic adjustment. (*Default:* `t`)

Options

```

\documentclass[11pt,a4paper]{article}

\usepackage{marginalia}
\marginaliasetup{
  ysep page top=0mm,
  style recto outer=\raggedright\footnotesize,
  style recto inner=\raggedleft\footnotesize,
  width=30mm,
}

\begin{document}

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod%
\marginalia[valign=t]{\large\ttfamily valign=t}\\The baseline of this marginal note
  is the baseline of the its top line.}%
\marginalia[valign=b,pos=reverse]{\large\ttfamily valign=b}\\The  baseline of this
  marginal note is the baseline of its bottom line.}

\vspace{30mm}

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi%
\marginalia[valign=c,pos=reverse]{\large\ttfamily valign=c}\\The baseline of this
  marginal note is midway between the top and bottom of its content.}%
\marginalia[valign=m]{\large\ttfamily valign=m}\\The baseline of this marginal note
  is midway between the baseline of its top line and the baseline of its bottom line.}

\end{document}

```

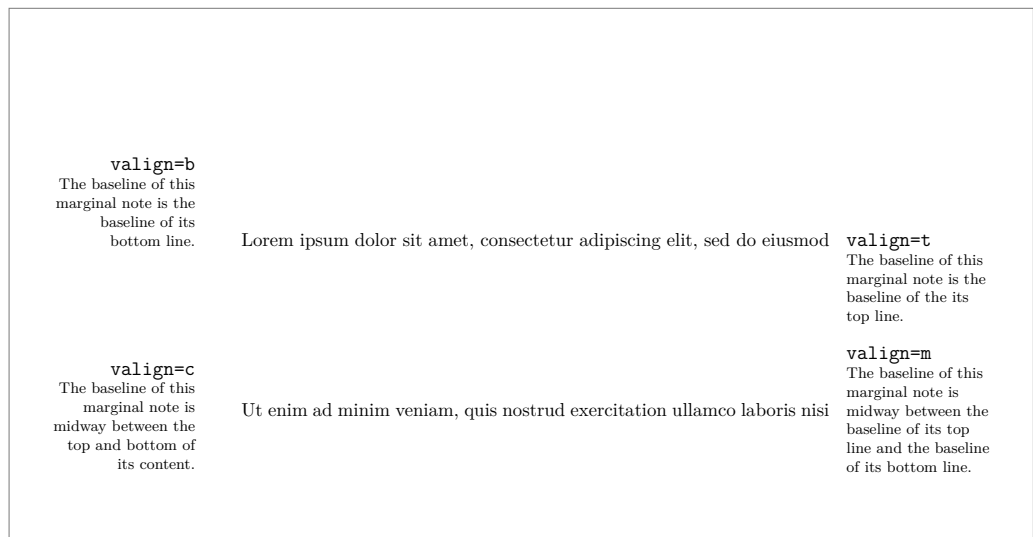


Figure 6.1: A demonstration of the various values of `valign`.

Options

The option `valign=c` may be useful if the marginal content contains a picture, which would sit on a single oversized line. Conversely, the option `valign=m` may be useful if one wants textual alignment with the body text.

yshift The key `yshift` is used to shift the default position of the marginal content item up (positive) or down (negative) from its normal position, which is to have its baseline aligned with the baseline of the callout position. It must be set to a valid dimension. Note that if `type=normal`, then the vertical position may be adjusted from that specified by `yshift`. If this is not desired, specify a different `type`. (*Default: 0pt*).

ysep These keys specify the minimum vertical separation above and below an item of marginal content (see [Figure 6.2](#)).

ysep above **ysep above:** the minimum vertical separation between an item and the one above. (*Default:* value of `\marginparpush` at `\begin{document}`)

ysep below **ysep below:** the minimum vertical separation between an item and the one below. (*Default:* value of `\marginparpush` at `\begin{document}`)

ysep page top **ysep page top:** the minimum vertical separation between an item and top of the page. (*Default:* margin above main textblock at `\begin{document}`)

ysep page bottom **ysep page bottom:** the minimum vertical separation between an item and bottom of the page. (*Default:* margin below main textblock at `\begin{document}`)

ysep above below: is a shorthand for setting both `ysep above` and `ysep below` simultaneously to the same value.

ysep: is a shorthand for setting all of these keys simultaneously to the same value. Each of these keys must be set to a valid dimension.

ysep page top margin These keys automatically set vertical separation between an item of marginal content and the top and bottom of the page to match the main textblock.

ysep page bottom margin **ysep page top margin:** Automatically set `ysep page top` to match the margins above the main textblock; to be precise, `ysep page top` is set to the value of $1\text{in} + \text{\voffset} + \text{\topmargin} + \text{\headheight} + \text{\headsep}$.

ysep page top bottom margin **ysep page bottom margin:** Automatically set `ysep page bottom` to match the margins above the main textblock; to be precise, `ysep page bottom` is set to the value of $\text{\paperheight} - (1\text{in} + \text{\voffset} + \text{\topmargin} + \text{\headheight} + \text{\headsep}) - \text{\textheight}$.

ysep page top bottom margin: Automatically set `ysep page top` and `ysep page bottom` to match the margins above and below the main textblock; has the same effect as specifying `ysep page top margin` and `ysep page bottom margin` separately.

None of these keys takes a value. Note that if the sizes of the top and bottom margins are changed, the values of `ysep page top` and `ysep page bottom` do not change automatically, even if these options have been used. The options can of course be used immediately after the new margins have been set.

6.4 Appearance

An item of marginal content that appears in the inner margin might be narrower than one that appears in the outer margin, and an item appearing in the outer margin of a recto page might be set ragged right, while an item appearing in the outer margin of a verso page might be set ragged left. And since it is not known where an item will appear until the page is assembled, the keys in this subsection, dealing with the width and style of an item, have variants that apply depending on where the item appears on the page.

Options

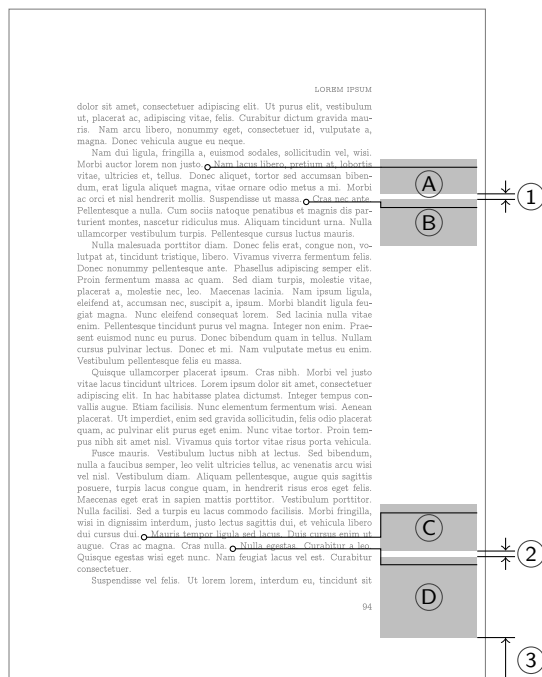


Figure 6.2: (Illustration of `ysep`) The length ① is at least the value of `ysep below` specified (locally or globally) for marginal content item ① and at least the value of `ysep above` specified for item ②. In this example diagram, ② has been automatically moved down from its natural position to maintain the required distance. Similarly, the length ② is at least the value of `ysep below` specified for ③ and at least the value of `ysep above` specified for ④, and the length ③ is at least the value of `ysep page bottom` specified for ④. In this example, to maintain the required distances, ③ and ④ have been automatically moved (respectively) up and down from their natural positions.

<code>width</code>	These keys specify the width of the an item of marginal content (or, more precisely,
<code>width outer</code>	the <code>\hsize</code> of the box into which the item is typeset). Which width is chosen will depend
<code>width inner</code>	on the where the item is typeset. The terminology is as in Figure 7.1 .
<code>width between</code>	<code>width recto outer</code> : used for an item in the outer margin of a recto page.
<code>width recto outer</code>	<code>width recto inner</code> : used for an item in the inner margin of a recto page.
<code>width recto inner</code>	<code>width verso outer</code> : used for an item in the outer margin of a verso page.
<code>width verso outer</code>	<code>width verso inner</code> : used for an item in the inner margin of a verso page.
<code>width verso inner</code>	<code>width right between</code> : used for an item set from the right column and placed be-
<code>width right between</code>	tween the columns.
<code>width left between</code>	<code>width left between</code> : used for an item set from the right column and placed between
	the columns.
	<code>width outer</code> : a shorthand for setting the keys <code>width recto outer</code> and <code>width verso</code>
	<code>outer</code> simultaneously to the same value.
	<code>width inner</code> : a shorthand for setting the keys <code>width recto inner</code> and <code>width verso</code>
	<code>inner</code> simultaneously to the same value.
	<code>width between</code> : a shorthand for setting the keys <code>width right between</code> and <code>width</code>
	<code>left between</code> simultaneously to the same value.
	<code>width</code> : a shorthand for setting all of these keys simultaneously.

Options

Table 2: Summary of keys affecting the appearance of the marginal content. (For keys affecting the type or position of marginal content, see [Table 1](#).) All keys can be set using `\marginaliasetup` or passed in the optional argument to `\marginalia`.

Key name	Value	Default
<code>width</code>	Dimension	<code>\marginparwidth</code>
<code>width outer</code>	Dimension	<code>\marginparwidth</code>
<code>width inner</code>	Dimension	<code>\marginparwidth</code>
<code>width between</code>	Dimension	<code>\marginparwidth</code>
<code>width recto outer</code>	Dimension	<code>\marginparwidth</code>
<code>width recto inner</code>	Dimension	<code>\marginparwidth</code>
<code>width verso outer</code>	Dimension	<code>\marginparwidth</code>
<code>width verso inner</code>	Dimension	<code>\marginparwidth</code>
<code>width right between</code>	Dimension	<code>\marginparwidth</code>
<code>width left between</code>	Dimension	<code>\marginparwidth</code>
<code>style</code>	L ^A T _E X code	[Empty]
<code>style recto outer</code>	L ^A T _E X code	[Empty]
<code>style recto inner</code>	L ^A T _E X code	[Empty]
<code>style verso outer</code>	L ^A T _E X code	[Empty]
<code>style verso inner</code>	L ^A T _E X code	[Empty]
<code>style right between</code>	L ^A T _E X code	[Empty]
<code>style left between</code>	L ^A T _E X code	[Empty]
<code>mark</code>	L ^A T _E X code	[Empty]
<code>mark recto outer</code>	L ^A T _E X code	[Empty]
<code>mark recto inner</code>	L ^A T _E X code	[Empty]
<code>mark verso outer</code>	L ^A T _E X code	[Empty]
<code>mark verso inner</code>	L ^A T _E X code	[Empty]
<code>mark right between</code>	L ^A T _E X code	[Empty]
<code>mark left between</code>	L ^A T _E X code	[Empty]

(The shorthands `width outer` and `width inner` exist because page geometry is usually symmetrical between recto and verso pages as regards outer and inner margins. The shorthand `width between` exists because the space between columns, if used at all for marginal content, will often be shared equally.) Each of these keys must be set to a valid dimension. (*Default:* value of `\marginparwidth` at `\begin{document}`})

`style` These keys specify the style with which an item of marginal content is typeset.

`style recto outer` Which style is chosen will depend on where the item is typeset. The terminology is as in [Figure 7.1](#).

`style recto inner`

`style verso outer` **style recto outer:** used for an item in the outer margin of a recto page.

`style verso inner` **style recto inner:** used for an item in the inner margin of a recto page.

`style right between` **style verso outer:** used for an item in the outer margin of a verso page.

`style left between` **style verso inner:** used for an item in the inner margin of a verso page.

style right between: used for an item set from the right column between the columns.

style left between: used for an item set from the right column between the columns.

style: a shorthand for setting all of these keys simultaneously.

Placement Each of these keys should be set to L^AT_EX code that specifies the style. (*Default:* [Empty])

These keys specify code to typeset something alongside the line where the call to `\marginalia` appears, on the same side on which the marginal content is placed. Roughly speaking, they can be set to any L^AT_EX code, from a single symbol to a TikZ picture. Which code is chosen will depend on where the marginal item is typeset. If the marginal item is placed on the right of the text, the `mark` code will be executed in an `\rlap` flush with the right end of the line; if the marginal item is placed on the left of the text, the `mark` code will be executed in an `\llap` flush with the left end of the line. These could be used to place arrowheads or more complicated indicators associating a marginal item with the text; see [Figure 6.3](#)

The terminology for the various `mark` options is as in [Figure 7.1](#).

mark recto outer: used for an item in the outer margin of a recto page.

mark recto inner: used for an item in the inner margin of a recto page.

mark verso outer: used for an item in the outer margin of a verso page.

mark verso inner: used for an item in the inner margin of a verso page.

mark right between: used for an item set from the right column between the columns.

mark left between: used for an item set from the right column between the columns.

mark: a shorthand for setting all of these keys simultaneously.

Each of these keys should be set to L^AT_EX code. (*Default:* [Empty])

6.5 Warning message configuration

The options described in this subsection have no effect on the placement of marginal content, but simply affect the number of problems or changes that are listed at `\end{document}`. Note that each of them can be set to 0 to suppress the list entirely (although warnings will still be emitted).

`max problem count` These options specify the maximum number of placement problems, the maximum number of changes in marginal content item data, and the maximum number of changes in page data to list. (*Default:* 40, 10, 10, respectively.)

7 Placement

The placement of an item of marginal content depends on where the call to `\marginalia` appears in the finished document. Both horizontal and vertical placement can be complicated.

7.1 Horizontal placement

To understand the horizontal placement, first recall some terminology: a recto page is an odd-numbered page in two-sided mode, or any page in one-sided mode; a verso page is an even-numbered page in two-sided mode. The description in the paragraphs that follow is summarized in [Figure 7.1](#).

In one-column mode, marginal content is placed by default in the outer margin: right on recto pages, left on verso pages. If `pos=reverse` is applied, it is placed in the inner margin: left on recto pages, right on verso pages.

In two-column mode, the default placement is next to the column in which the call to `\marginalia` appears, on the side opposite to the other column. Thus, if the call to `\marginalia` was in the left column, the marginal content item is placed by default on the

Placement

```

\documentclass[11pt,a4paper]{article}

\usepackage{marginalia}
\marginaliasetup{
  xsep=5mm, style=\raggedright\sffamily,
}

\usepackage{tikz}
\newcommand\drawarrow{%
  \tikz[remember picture,overlay]
    \draw[->,thick] (arrowstart) -- ++(-1.5mm,0) |- (.5mm,.5ex);%
}
\newcommand\savearrowstart{%
  \tikz[remember picture,overlay] \coordinate (arrowstart) at (-.5mm,.5ex);%
}

\begin{document}

Lorem ipsum dolor sit amet,%
\marginalia[mark={~$\triangleleft$}]{A marginal note. (Extra text to push down the
note below.)} consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat.%
\marginalia[mark={\drawarrow}]{\savearrowstart Another marginal note.}
Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur.

\end{document}

```

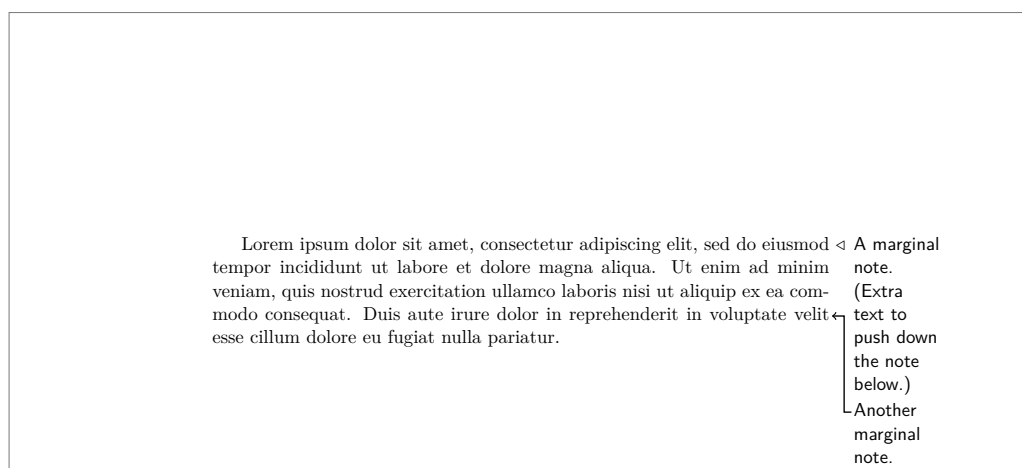


Figure 6.3: A demonstration of the `mark` option, in particular showing how it can be used to draw a TikZ arrow from a (moved) note to the relevant line. Notes: (1) The code in the marginal content is processed *before* the mark, so the `mark` TikZ code refers to a coordinate defined in the marginal content, not the other way around. (2) The code must be compiled *three* times to give this output, because the `arrowstart` coordinate is not placed correctly until after `marginalia` has placed the second marginal note on the second run. Thus one more run is necessary for TikZ to draw the arrow correctly.

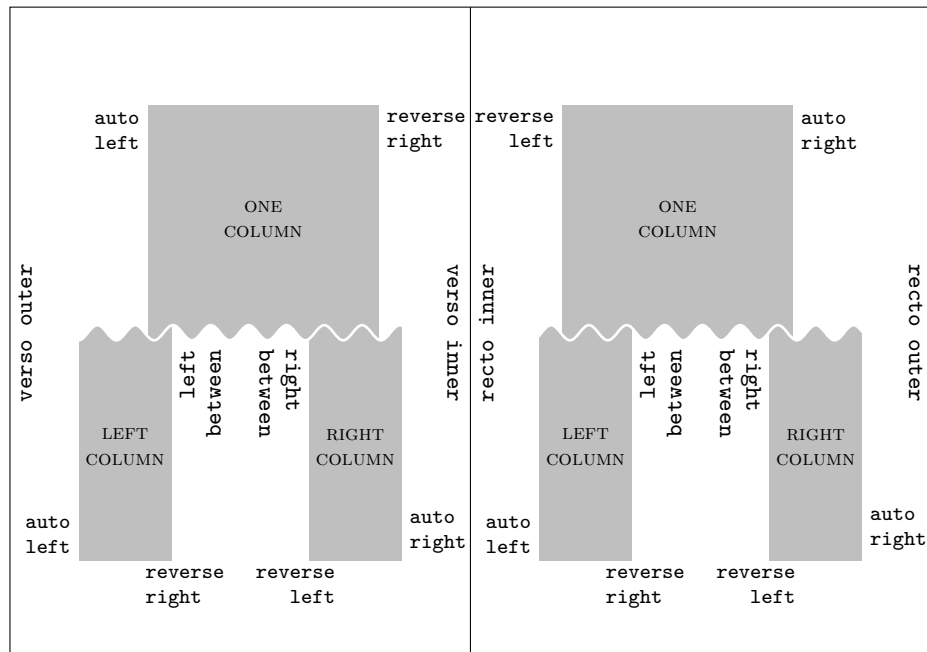


Figure 7.1: Summary of the positioning of marginal content using `pos`, and terminology used in `width` and `style` keys, on recto and verso pages, in both one-column and two-column mode.

`left`: on a recto page, the inner margin, on a verso page, the outer margin. If `pos=reverse` is applied, it is placed between the two columns, adjacent to the left column. If the call to `\marginalia` was in the right column, the item is placed by default on the right: on a recto page, the outer margin, on a verso page, the inner margin. If `pos=reverse` is applied, it is placed between the two columns, adjacent to the right column.

`pos=left` specifies that the item is to be placed on the left of the text block or column containing the call to `\marginalia`.

`pos=right` similarly specifies that the item is to be placed on the right of the text block or column containing the call to `\marginalia`.

`marginalia` determines in which column the call to `\marginalia` was made using its horizontal position. As discussed in the description of key `column`, there are situations where this can go wrong and which necessitate a manual specification of a particular column.

7.2 Vertical placement

`marginalia` tries by default to place the each item of marginal content with its baseline shifted by the value of `yshift` (by default, 0pt) from the baseline where `\marginalia` was called. The actual vertical placement is calculated by the procedure described below, carried out for the items appearing in a particular horizontal location. (As shown in Figure 7.1, in one-column mode the possible locations are in outer and inner margins; in two-column mode the possible locations are the outer and inner margins and on the left and right sides of the space between the columns.) A *clash* exists when two items

Usage notes

are closer than specified by `ysep below` for the upper item or `ysep above` for the lower item, whichever is greater.

For the items in each horizontal location, the procedure is as follows:

1. Place the items appearing in a given horizontal location on the page into a list.
2. Set the vertical shift of each item to the one specified by `yshift`.
3. For each `type=optfixed` item, if it clashes with any `type=fixed` item, delete it from the list of items that appear on the page.
4. Sort the list by the position of the call to `\marginalia`, top-to-bottom, left-to-right, breaking ties by the order of calls. (Because of floats, footnotes, etc., the sorted order of the list is not necessarily the same as the order of appearance of `\marginalia` commands in the source code.)
5. Pass through the list of items in sorted order. For each `type=normal` item, if necessary shift it in a negative (downward) direction so that it (1) does not reach closer to the top of the page than specified by `ysep page top`, and (2) does not clash with the previous (above) item. (After this stage, it is possible for an assigned vertical shift to push a `type=normal` item off the bottom of the page.)
6. Pass through the list of items in the reverse of the sorted order. For each `type=normal` item, if necessary shift it in a positive (upward) direction so that it (1) does not reach closer to the bottom of the page than specified by `ysep page bottom`, and (2) does not clash with the next (below) item.

During this process, it may be found that it is impossible to prevent clashes or items reaching beyond the limits (e.g. fixed items clash with each other; a fixed item conflicts with `ysep page top` or `ysep page bottom`, or there are simply too many items of marginal content to fit (in which case, the top of some of them will be above the limit specified by `ysep page top` or will clash with fixed items)). In these cases, warnings are issued at the end of the Lua $\text{\LaTeX}$ run.

8 Usage notes

`marginalia` requires a minimum of two Lua $\text{\LaTeX}$ runs, and often more, to place items of marginal content correctly. On the first pass, information about items, including their vertical size, is written to the `.aux` file, and this information is used to position them correctly on the next run. However, because `width` and `style` have variants dependent on the margin in which the item is placed, an item may only be typeset at the correct size on this second run. Thus the vertical size of the item may have changed and so the information written to the `.aux` file on the previous run may be out of date. In this case a third run may be needed for correct placement.

More runs may be needed if the position of the call to `\marginalia` changes between runs. (For example, if `\marginalia` is used to set numbered sidenotes with per-page numbering, the number may change between runs and the small difference in widths can change line breaks and page/column breaks.) Provided that the main text stabilizes, the placement of items using `\marginalia` should be correct two runs later.

At the end of the Lua $\text{\LaTeX}$ run, `marginalia` reports any problems encountered in the vertical placement of items (as described at the end of [Subsection 7.2](#)). These problems are based on calculations made on the basis of information from the previous written

to the `.aux` file on the previous run, and may not arise if item positions or sizes (i.e. height or depth) have changed. `marginalia` also reports any changes in positions or sizes compared to the previous run.

In these reports, a page number refers to a visible page number if it is prefixed with ‘p’; it otherwise refers to the absolute page number of the output.

9 Incompatibilities

Using `marginalia` alongside `\marginpar` or packages like `mparhak`, `marginnote`, `marginfix`, or `marginfit` should not produce any errors, but `marginalia` will ignore marginal content not created using `\marginalia`; for example, an item of marginal content created using `\marginalia` might overlap with one created using `\marginpar`.

10 Limitations

As noted in the introduction, `marginalia` was originally written to typeset a particular kind of book. It thus has several limitations. Three of these are:

Lua $\LaTeX$ Only Most of the code for deciding the placement of items of marginal content is written in Lua. In principle, it could be replaced with a pure $\LaTeX$ solution.

No support for ‘moving past’ fixed items The adjustment of vertical positions will never cause a `type=normal` item to be shifted past a `type=fixed` one, even when there is space on the other side. It may be desirable to have this available as an option.

No support for nested content items Nesting might be desirable for typesetting editions of manuscripts which sometimes contain marginal glosses, and then glosses upon those glosses.

The lack of any built-in facility for producing (for example) numbered sidenotes is a conscious design choice. This is properly the concern of a command that merely uses `\marginalia` to place the notes correctly.

References

- [Bri04] R. Bringhurst. *The Elements of Typographic Style*. Hartley & Marks, version 3.0, 2004.
- [Cai24] A. J. Cain. *Form & Number: A History of Mathematical Beauty*. Lisbon, 2024. URL: https://archive.org/details/cain_formandnumber_ebook_large.
- [Cai25] A. J. Cain ‘`marginalia` at work: Running heads, float captions, citations, and small figures in the margins’. *TUGboat*. The Communications of the $\TeX$ Users Group. 46, no.1 (2025), pp.49–53. DOI: [10.47397/tb/46-1/tb142cain-marginalia](https://doi.org/10.47397/tb/46-1/tb142cain-marginalia)

11 Implementation (L^AT_EX package)

```

1 <*package>
2 <@@=marginalia>

```

11.1 Initial set-up

Package identification/version information.

```

3 \NeedsTeXFormat{LaTeX2e}[2020-02-02]
4 \ProvidesExplPackage{marginalia}{2026-07-28}{0.83.30}
5 {Non-floating marginal content for LuaLaTeX}

```

Check that Lua_TE_X is in use.

```

6 \sys_if_engine luatex:F
7 {
8   \msg_new:nnn{marginalia}{lualatex_required}
9   {LuaLaTeX-required.~Package-loading-will-abort.}
10  \msg_critical:nn{marginalia}{lualatex_required}
11 }

```

11.2 Tagging set-up

If L^AT_EX has tagging support, set up sockets if necessary and define `\__marginalia_tagging_socket:n` to be `\UseTaggingSocket`.

```

12 \ifundefined{UseTaggingSocket}
13 {
14   \cs_new:Npn \__marginalia_tagging_socket:n #1 {}
15 }
16 {
17   \str_if_exist:cF { l__socket_tagsupport/marginpar/begin_plug_str }
18   {
19     \socket_new:nn {tagsupport/marginpar/begin}{0}
20     \socket_new:nn {tagsupport/marginpar/end}{0}
21   }
22   \str_if_exist:cF { l__socket_tagsupport/para/restore_plug_str }
23   {
24     \socket_new:nn {tagsupport/para/restore}{0}
25   }
26   \cs_new:Npn \__marginalia_tagging_socket:n #1
27   {
28     \UseTaggingSocket{#1}
29   }
30 }

```

11.3 Auxiliary macro for dimension setting

`\__marginalia_set_dim:Nn` Set the dimension variable passed as the first parameter to value specified in second parameter at `begindocument` if used in the preamble, or immediately (since `begindocument` is a one-time hook) in the document.

```

31 \cs_new:Nn \__marginalia_set_dim:Nn
32 {
33   \hook_gput_code:nnn { begindocument } { marginalia/dim }
34   {
35     \dim_set:Nn #1 { #2 }
36   }
37 }

```

```

36     }
37 }

```

(End of definition for `\_marginalia\_set\_dim:Nn`.)

11.4 Auxiliary macros for setting options

`\_marginalia\_setup\_preamble:n` Macro used to set the configuration in the preamble. This only has effect at the outer group level: inside a group, options should be confined to that group, so adding the options that use the `begindocument` hook (via `\_marginalia\_set\_dim:Nn`) should have no effect. And since `\marginalia` cannot be used in the preamble, setting other options inside a group in the preamble is pointless.

```

38 \cs_new:Npn \_marginalia\_setup\_preamble:n #1
39 {
40   \int_if_zero:N\T{ \currentgrouplevel }
41   {
42     \keys\_set:nn{marginalia}{ #1 }
43   }
44 }

```

(End of definition for `\_marginalia\_setup\_preamble:n`.)

`\_marginalia\_setup\_body:n` Macro used to set the configuration in the document body.

```

45 \cs_new:Npn \_marginalia\_setup\_body:n #1
46 {
47   \keys\_set:nn{marginalia}{ #1 }
48 }

```

(End of definition for `\_marginalia\_setup\_body:n`.)

`\_marginalia\_setup:n` The macro `\_marginalia\_setup:n` is defined to be `\_marginalia\_setup\_preamble:n` initially and is redefined to `\_marginalia\_setup\_body:n` at `begindocument/end`.

```

49 \cs\_set\_eq:NN\_marginalia\_setup:n\_marginalia\_setup\_preamble:n
50 \hook\_gput\_code:nnn{ begindocument/end }{ marginalia/setup }
51 {
52   \cs\_set\_eq:NN\_marginalia\_setup:n\_marginalia\_setup\_body:n
53 }

```

(End of definition for `\_marginalia\_setup:n`.)

11.5 Options

Set up the key–value options and the variables in which the settings will be stored.

11.5.1 Type

`\l__marginalia_type_int` A key to store the type of the marginal content item. The setting is held in an integer variable: 1 = normal, 2 = fixed, 3 = optfixed.

```

54 \int_new:N\l__marginalia_type_int
55 \keys_define:nn { marginalia }
56 {
57   type .choices:nn = {normal,fixed,optfixed}{
58     \int_set:Nn\l__marginalia_type_int{\l_keys_choice_int}
59   },
60   type .initial:n = normal,
61 }

```

(End of definition for \l__marginalia_type_int.)

11.5.2 Horizontal placement

`\l__marginalia_pos_int` A key to store the specified position of the marginal content item. The setting is held in an integer variable: 1 = auto, (the outer margin in one-column mode; left margin in left column, right margin in right column in two-column mode) 2 = reverse (inner margin in one-column mode; between the columns in two-column mode), 3 = left, 4 = right, 5 = nearest.

```

62 \int_new:N\l__marginalia_pos_int
63 \keys_define:nn { marginalia }
64 {
65   pos .choices:nn = {auto,reverse,left,right,nearest}{
66     \int_set:Nn\l__marginalia_pos_int{\l_keys_choice_int}
67   },
68   pos .initial:n = auto
69 }

```

(End of definition for \l__marginalia_pos_int.)

`\l__marginalia_column_int` A key to force the marginal content item to be treated in one-column mode or as being set from the left or right column. The setting is held in an integer variable: -1 = auto (automatic), 0 = one (one-column mode), 1 = left (left column) 2 = right (right column).

```

70 \int_new:N\l__marginalia_column_int
71 \keys_define:nn { marginalia }
72 {
73   column .choices:nn = {auto,one,left,right}{
74     \int_set:Nn\l__marginalia_column_int{\l_keys_choice_int-2}
75   },
76   column .initial:n = auto,
77 }

```

(End of definition for \l__marginalia_column_int.)

`\l__marginalia_xsep_recto_outer_dim`
`\l__marginalia_xsep_recto_inner_dim`
`\l__marginalia_xsep_verso_outer_dim`
`\l__marginalia_xsep_verso_inner_dim`
`\l__marginalia_xsep_right_between_dim`
`\l__marginalia_xsep_left_between_dim`

Dimension keys to hold the separation between the marginal content item and the main text, which can be dependent on where it appears on the page.

```

78 \dim_new:N\l__marginalia_xsep_recto_outer_dim
79 \dim_new:N\l__marginalia_xsep_recto_inner_dim
80 \dim_new:N\l__marginalia_xsep_verso_outer_dim
81 \dim_new:N\l__marginalia_xsep_verso_inner_dim

```

```

82 \dim_new:N\l__marginalia_xsep_right_between_dim
83 \dim_new:N\l__marginalia_xsep_left_between_dim
84 \keys_define:nn { marginalia }
85 {
86   xsep~recto~outer .code:n
87     = \__marginalia_set_dim:Nn\l__marginalia_xsep_recto_outer_dim{#1},
88   xsep~recto~inner .code:n
89     = \__marginalia_set_dim:Nn\l__marginalia_xsep_recto_inner_dim{#1},
90   xsep~verso~outer .code:n
91     = \__marginalia_set_dim:Nn\l__marginalia_xsep_verso_outer_dim{#1},
92   xsep~verso~inner .code:n
93     = \__marginalia_set_dim:Nn\l__marginalia_xsep_verso_inner_dim{#1},
94   xsep~right~between .code:n
95     = \__marginalia_set_dim:Nn\l__marginalia_xsep_right_between_dim{#1},
96   xsep~left~between .code:n
97     = \__marginalia_set_dim:Nn\l__marginalia_xsep_left_between_dim{#1},
98   xsep .code:n = {
99     \keys_set:nn{ marginalia }{
100       xsep~recto~outer=#1,
101       xsep~recto~inner=#1,
102       xsep~verso~outer=#1,
103       xsep~verso~inner=#1,
104       xsep~right~between=#1,
105       xsep~left~between=#1,
106     }
107   },
108   xsep~outer .code:n = {
109     \keys_set:nn{ marginalia }{
110       xsep~recto~outer=#1,
111       xsep~verso~outer=#1,
112     }
113   },
114   xsep~inner .code:n = {
115     \keys_set:nn{ marginalia }{
116       xsep~recto~inner=#1,
117       xsep~verso~inner=#1,
118     }
119   },
120   xsep~between .code:n = {
121     \keys_set:nn{ marginalia }{
122       xsep~right~between=#1,
123       xsep~left~between=#1,
124     }
125   },
126   xsep .initial:n = \marginparsep,
127 }

```

(End of definition for \l__marginalia_xsep_recto_outer_dim and others.)

11.5.3 Vertical placement

`\l__marginalia_valign_int` A key to store the vertical alignment of the marginal content item. The setting is held in an integer variable: 1 = t (aligned at the baseline of the topmost line of the item), 2 = b (aligned at the baseline of the bottommost line of the item).

```

128 \int_new:N\l__marginalia_valign_int
129 \keys_define:nn { marginalia }
130 {
131   valign .choices:nn = {t,b,c,m}{
132     \int_set_eq:NN\l__marginalia_valign_int\l_keys_choice_int
133   },
134   valign .initial:n = t,
135 }

```

(End of definition for \l__marginalia_valign_int.)

\l__marginalia_default_yshift_dim Dimension key to hold the default vertical shift of the marginal content item from its natural position.

```

136 \keys_define:nn { marginalia }
137 {
138   yshift .dim_set:N = \l__marginalia_default_yshift_dim,
139   yshift .initial:n = 0pt,
140 }

```

(End of definition for \l__marginalia_default_yshift_dim.)

__marginalia_margin_top: These macros are simply the calculations necessary for the space above and below the main textblock. They are simply a convenience to avoid specifying the calculation twice in the definition of the ysep keys.

```

141 \cs_new:Npn \__marginalia_margin_top:
142 {
143   1in + \voffset + \topmargin + \headheight + \headsep
144 }
145 \cs_new:Npn \__marginalia_margin_bottom:
146 {
147   \pageheight - 1in - \voffset - \topmargin - \headheight - \headsep
148   - \textheight
149 }

```

(End of definition for __marginalia_margin_top: and __marginalia_margin_bottom:.)

\l__marginalia_ysep_above_dim Dimension keys to hold the the minimum vertical spacing between a marginal content item and (respectively) the item above, the item below, the page top, and the page bottom.

```

150 \dim_new:N\l__marginalia_ysep_above_dim
151 \dim_new:N\l__marginalia_ysep_below_dim
152 \dim_new:N\l__marginalia_ysep_page_top_dim
153 \dim_new:N\l__marginalia_ysep_page_bottom_dim
154 \keys_define:nn { marginalia }
155 {
156   ysep~above .code:n
157     = \__marginalia_set_dim:Nn\l__marginalia_ysep_above_dim{#1},
158   ysep~below .code:n
159     = \__marginalia_set_dim:Nn\l__marginalia_ysep_below_dim{#1},
160   ysep~page~top .code:n
161     = \__marginalia_set_dim:Nn\l__marginalia_ysep_page_top_dim{#1},
162   ysep~page~bottom .code:n
163     = \__marginalia_set_dim:Nn\l__marginalia_ysep_page_bottom_dim{#1},
164   ysep~above~below .code:n = {

```

```

165     \keys_set:nn{ marginalia }{
166       ysep~below=#1,
167       ysep~above=#1,
168     }
169   },
170   ysep .code:n = {
171     \keys_set:nn{ marginalia }{
172       ysep~below=#1,
173       ysep~above=#1,
174       ysep~page~top=#1,
175       ysep~page~bottom=#1,
176     }
177   },
178   ysep~page~top~margin .code:n = {
179     \keys_set:nn{ marginalia }{
180       ysep~page~top
181       = \__marginalia_margin_top:
182     }
183   },
184   ysep~page~bottom~margin .code:n = {
185     \keys_set:nn{ marginalia }{
186       ysep~page~bottom
187       = \__marginalia_margin_bottom:
188     }
189   },
190   ysep~page~top~bottom~margin .code:n = {
191     \keys_set:nn{ marginalia }{
192       ysep~page~top~margin,
193       ysep~page~bottom~margin,
194     }
195   },
196   ysep~above~below .initial:n = \marginparpush,
197   ysep~page~top .initial:n = \__marginalia_margin_top:,
198   ysep~page~bottom .initial:n = \__marginalia_margin_bottom:,
199 }

```

(End of definition for \l__marginalia_ysep_above_dim and others.)

11.5.4 Appearance

`\l__marginalia_width_recto_outer_dim` Dimension keys to hold the width of the marginal content item, which can be dependent on where it appears on the page.

```

200 \dim_new:N\l__marginalia_width_recto_outer_dim
201 \dim_new:N\l__marginalia_width_recto_inner_dim
202 \dim_new:N\l__marginalia_width_verso_outer_dim
203 \dim_new:N\l__marginalia_width_verso_inner_dim
204 \dim_new:N\l__marginalia_width_right_between_dim
205 \dim_new:N\l__marginalia_width_left_between_dim
206 \keys_define:nn { marginalia }
207 {
208   width~recto~outer .code:n
209   = \__marginalia_set_dim:Nn\l__marginalia_width_recto_outer_dim{#1},
210   width~recto~inner .code:n
211   = \__marginalia_set_dim:Nn\l__marginalia_width_recto_inner_dim{#1},

```



```

212 width~verso~outer .code:n
213   = \__marginalia_set_dim:Nn\l__marginalia_width_verso_outer_dim{#1},
214 width~verso~inner .code:n
215   = \__marginalia_set_dim:Nn\l__marginalia_width_verso_inner_dim{#1},
216 width~right~between .code:n
217   = \__marginalia_set_dim:Nn\l__marginalia_width_right_between_dim{#1},
218 width~left~between .code:n
219   = \__marginalia_set_dim:Nn\l__marginalia_width_left_between_dim{#1},
220 width .code:n = {
221   \keys_set:nn{ marginalia }{
222     width~recto~outer=#1,
223     width~recto~inner=#1,
224     width~verso~outer=#1,
225     width~verso~inner=#1,
226     width~right~between=#1,
227     width~left~between=#1,
228   }
229 },
230 width~outer .code:n = {
231   \keys_set:nn{ marginalia }{
232     width~recto~outer=#1,
233     width~verso~outer=#1,
234   }
235 },
236 width~inner .code:n = {
237   \keys_set:nn{ marginalia }{
238     width~recto~inner=#1,
239     width~verso~inner=#1,
240   }
241 },
242 width~between .code:n = {
243   \keys_set:nn{ marginalia }{
244     width~right~between=#1,
245     width~left~between=#1,
246   }
247 },
248 width .initial:n = \marginparwidth,
249 }

```

(End of definition for \l__marginalia_width_recto_outer_dim and others.)

```

\l__marginalia_style_recto_outer_tl
\l__marginalia_style_recto_inner_tl
\l__marginalia_style_verso_outer_tl
\l__marginalia_style_verso_inner_tl
\l__marginalia_style_right_between_tl
\l__marginalia_style_left_between_tl

```

Token list keys to hold the style with which a marginal content item is typeset, which can be dependent on where it appears on the page.

```

250 \keys_define:nn { marginalia }
251 {
252   style~recto~outer .tl_set:N = \l__marginalia_style_recto_outer_tl,
253   style~recto~inner .tl_set:N = \l__marginalia_style_recto_inner_tl,
254   style~verso~outer .tl_set:N = \l__marginalia_style_verso_outer_tl,
255   style~verso~inner .tl_set:N = \l__marginalia_style_verso_inner_tl,
256   style~right~between .tl_set:N = \l__marginalia_style_right_between_tl,
257   style~left~between .tl_set:N = \l__marginalia_style_left_between_tl,
258   style .code:n = {
259     \keys_set:nn{ marginalia }{
260       style~recto~outer=#1,

```

```

261     style~recto~inner=#1,
262     style~verso~outer=#1,
263     style~verso~inner=#1,
264     style~right~between=#1,
265     style~left~between=#1,
266   }
267 },
268 style .initial:n = {},
269 }

```

(End of definition for \l__marginalia_style_recto_outer_tl and others.)

\l__marginalia_mark_recto_outer_tl Token list keys to hold code for a mark to be placed adjacent to the line where the call
 \l__marginalia_mark_recto_inner_tl to\marginalia is located, on the same side as the marginal content item.
 \l__marginalia_mark_verso_outer_tl
 \l__marginalia_mark_verso_inner_tl
 \l__marginalia_mark_right_between_tl
 \l__marginalia_mark_left_between_tl

```

270 \keys_define:nn { marginalia }
271 {
272   mark~recto~outer .tl_set:N = \l__marginalia_mark_recto_outer_tl,
273   mark~recto~inner .tl_set:N = \l__marginalia_mark_recto_inner_tl,
274   mark~verso~outer .tl_set:N = \l__marginalia_mark_verso_outer_tl,
275   mark~verso~inner .tl_set:N = \l__marginalia_mark_verso_inner_tl,
276   mark~right~between .tl_set:N = \l__marginalia_mark_right_between_tl,
277   mark~left~between .tl_set:N = \l__marginalia_mark_left_between_tl,
278   mark .code:n = {
279     \keys_set:nn{ marginalia }{
280       mark~recto~outer=#1,
281       mark~recto~inner=#1,
282       mark~verso~outer=#1,
283       mark~verso~inner=#1,
284       mark~right~between=#1,
285       mark~left~between=#1,
286     }
287   },
288   mark .initial:n = {},
289 }

```

(End of definition for \l__marginalia_mark_recto_outer_tl and others.)

11.5.5 Warning message configuration

\l__marginalia_max_problem_int Integer keys to control the number of changes to be output
 \l__marginalia_max_page_data_change_int
 \l__marginalia_max_item_data_change_int

```

290 \keys_define:nn { marginalia }
291 {
292   max~problem~count .int_set:N = \l__marginalia_max_problem_int,
293   max~problem~count .initial:n= 40,
294   max~page~data~change~count .int_set:N = \l__marginalia_max_page_data_change_int,
295   max~page~data~change~count .initial:n = 10,
296   max~item~data~change~count .int_set:N = \l__marginalia_max_item_data_change_int,
297   max~item~data~change~count .initial:n = 10,
298 }

```

(End of definition for \l__marginalia_max_problem_int, \l__marginalia_max_page_data_change_int, and \l__marginalia_max_item_data_change_int.)

11.6 Lua backend and interface

Load the Lua backend.

```
299 \lua_now:n{ require('marginalia') }
```

The following 9 macros interface between L^AT_EX and Lua code. Each control sequence `\_marginalia_lua_XYZ` simply calls the corresponding Lua function `marginalia.XYZ`.

The first 8 macros do not require expansion of parameters: they either have none, or process data not containing control sequences (read from the `.aux` file); hence `\lua_now:n` is used.

```
\_marginalia_lua_store_default_page_data:
\_marginalia_lua_store_page_data:n
\_marginalia_lua_check_page_data:n
\_marginalia_lua_store_item_data:n
\_marginalia_lua_check_item_data:n
\_marginalia_lua_compute_items:
\_marginalia_lua_write_problem_report:
\_marginalia_lua_write_item_change_report:

300 \cs_new:Npn\_marginalia_lua_store_default_page_data:
301 {
302   \lua_now:n{ marginalia.store_default_page_data() }
303 }
304 \cs_new:Npn\_marginalia_lua_store_page_data:n #1
305 {
306   \lua_now:n{ marginalia.store_page_data('#1') }
307 }
308 \cs_new:Npn\_marginalia_lua_check_page_data:n #1
309 {
310   \lua_now:n{ marginalia.check_page_data('#1') }
311 }
312 \cs_new:Npn\_marginalia_lua_write_page_change_report:
313 {
314   \lua_now:n{ marginalia.write_page_change_report() }
315 }
316 \cs_new:Npn\_marginalia_lua_store_item_data:n #1
317 {
318   \lua_now:n{ marginalia.store_item_data('#1') }
319 }
320 \cs_new:Npn\_marginalia_lua_check_item_data:n #1
321 {
322   \lua_now:n{ marginalia.check_item_data('#1') }
323 }
324 \cs_new:Npn\_marginalia_lua_compute_items:
325 {
326   \lua_now:n{ marginalia.compute_items() }
327 }
328 \cs_new:Npn\_marginalia_lua_write_problem_report:
329 {
330   \lua_now:n{ marginalia.write_problem_report() }
331 }
332 \cs_new:Npn\_marginalia_lua_write_item_change_report:
333 {
334   \lua_now:n{ marginalia.write_item_change_report() }
335 }
```

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

The last macro will receive a control sequence parameter and so requires expansion; hence `\lua_now:e` is used.

```
336 \cs_new:Npn\_marginalia_lua_load_item_data:n #1
337 {
338   \lua_now:e{ marginalia.load_item_data('#1') }
```

```
339 }
```

(End of definition for _marginalia_lua_load_item_data:n.)

11.7 Processing data from the .aux file

`\marginalia@pagedata` This command is used to store version information in the .aux file. It currently does nothing, but may be used in future to avoid errors if changes are made in the format of the data written to the .aux file.

```
340 \cs_new:Npn \marginalia@version #1
341 {}
```

(End of definition for \marginalia@pagedata.)

`\marginalia@pagedata` This command is used to store page data in the .aux file.

```
342 \cs_new:Npn \marginalia@pagedata #1
343 {
344   \_marginalia_process_page_data:n{#1}
345 }
```

Initially `\_marginalia_process_page_data:n` is set to `\_marginalia_lua_store_page_data:n`. Thus, when the .aux file is read, `\marginalia@pagedata` will pass the page data to the Lua backend to be stored.

```
346 \cs_set_eq:NN
347   \_marginalia_process_page_data:n
348   \_marginalia_lua_store_page_data:n
```

(End of definition for \marginalia@pagedata.)

`\marginalia@itemdata` This command is used to store data for each marginal content item in the .aux file.

```
349 \cs_new:Npn \marginalia@itemdata #1
350 {
351   \_marginalia_process_item_data:n{#1}
352 }
```

(End of definition for \marginalia@itemdata.)

Initially `\_marginalia_process_item_data:n` is set to `\_marginalia_lua_store_item_data:n`. Thus, when the .aux file is read, `\marginalia@itemdata` will pass the item data to the Lua backend to be stored.

```
353 \cs_set_eq:NN
354   \_marginalia_process_item_data:n
355   \_marginalia_lua_store_item_data:n
```

At the `begindocument` hook, the .aux file has been read and closed. The Lua backend now stores the geometry and computes the vertical shift for each item. Then the handle for the main .aux file is stored for use in this package.

```
356 \hook_gput_code:nnn{ begindocument }{ marginalia/prepare }{
357   \_marginalia_lua_store_default_page_data:
358   \_marginalia_lua_compute_items:
359   \cs_set_eq:NN\l_marginalia_aux_iow\mainaux
360 }
```

The `enddocument/afterlastpage` hook is before the `.aux` file is read back, so this is where `\__marginalia_process_page_data:n` and `\__marginalia_process_item_data:n` are set, respectively, to `\__marginalia_lua_check_page_data:n` and `\__marginalia_lua_check_item_data:n`. Thus, when the `.aux` file is read back, `\marginalia@pagedata` and `\marginalia@itemdata` will pass data to the Lua backend to be checked for changes.

```

361 \hook_gput_code:nnn{enddocument/afterlastpage}{\ marginalia/check }{
362   \cs_set_eq:NN
363     \__marginalia_process_page_data:n
364     \__marginalia_lua_check_page_data:n
365   \cs_set_eq:NN
366     \__marginalia_process_item_data:n
367     \__marginalia_lua_check_item_data:n
368 }

```

`\__marginalia_write_reports:` Write reports if the `\marginalia` command has been used at least once, which is determined by checking whether `\g__marginalia_itemno_int` is non-zero.

```

369 \cs_new:Npn\__marginalia_write_reports:
370 {
371   \int_if_zero:nF{ \g__marginalia_itemno_int }
372   {
373     \__marginalia_lua_write_problem_report:
374     \__marginalia_lua_write_item_change_report:
375     \__marginalia_lua_write_page_change_report:
376   }
377 }

```

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

Use the `enddocument/info` hook to write the reports of changes and/or problems.

```

378 \hook_gput_code:nnn{ enddocument/info }{\ marginalia/report } {
379   \__marginalia_write_reports:
380 }

```

11.8 Writing version data to the `.aux` file

`\__marginalia_write_version:` This command will be used to write the package version to the `.aux` file.

```

381 \cs_new:Npn\__marginalia_write_version:
382 {
383   \iow_now:Ne\l__marginalia_aux_iow{
384     \token_to_str:N\marginalia@version{
385       \use:c{ver@marginalia.sty}
386     }
387   }
388 }

```

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

11.9 Writing page data to the `.aux` file

To compute the positions of marginal content items, certain page layout data is required. And since all the computation takes place at the beginning of the document, it is necessary to write this information to the `.aux` file.

`\g_marginalia_pagedatano_int` Global integer variable to index page data items written to the .aux file.

```
389 \int_new:N\g__marginalia_pagedatano_int
```

(End of definition for \g__marginalia_pagedatano_int.)

`\__marginalia_write_page_data:` This command will be used to write the current page data to the .aux file. It is initially defined to do nothing, so that the use of `\marginalianewgeometry` in the preamble does not cause errors (because the .aux file is not available for writing until `begindocument/end`).

```
390 \cs_set_eq:NN\__marginalia_write_page_data:\prg_do_nothing:
391 \cs_new:Npn\__marginalia_write_page_data_real:
392 {
393   \int_gincr:N\g__marginalia_pagedatano_int
394   \legacy_if:nTF{@twocolumn}
395     { \int_set:Nn\l_tmpa_int{2} }
396     { \int_set:Nn\l_tmpa_int{1} }
397   \iow_now:Ne\l__marginalia_aux_iow{
398     \token_to_str:N\marginalia@pagedata{
399       pagedatano=\int_value:w\g__marginalia_pagedatano_int,
400       abspageno=\int_eval:n{\g_shipout_readonly_int+1},
401       hoffset=\int_value:w\hoffset,
402       voffset=\int_value:w\voffset,
403       pageheight=\int_value:w\pageheight,
404       oddsidemargin=\int_value:w\oddsidemargin,
405       evensidemargin=\int_value:w\evensidemargin,
406       textwidth=\int_value:w\textwidth,
407       columncount=\int_value:w\l_tmpa_int,
408       columnwidth=\int_value:w\columnwidth,
409       columnsep=\int_value:w\columnsep,
410       twoside=\bool_to_str:n{\legacy_if_p:n{@twoside}},
411     }
412   }
413 }
```

At the `begindocument/end` hook, the .aux file has been opened for writing, and so the macro `\__marginalia_write_page_data:` is enabled and the initial page data is written out.

```
414 \hook_gput_code:nnn{ begindocument/end }{ marginalia/initial }
415 {
416   \__marginalia_write_version:
417   \cs_set_eq:NN
418     \__marginalia_write_page_data:
419     \__marginalia_write_page_data_real:
420   \__marginalia_write_page_data:
421 }
```

(End of definition for __marginalia_write_page_data:.)

11.10 Marginal content item processing

11.10.1 Variables

Variables set by $\text{\LaTeX}$.

`\g__marginalia_itemno_int` Global integer variable to index marginal content items.

```
422 \int_new:N\g__marginalia_itemno_int
```

(End of definition for \g__marginalia_itemno_int.)

`\l__marginalia_item_box` Box variable to hold the typeset marginal content item.

```
423 \box_new:N\l__marginalia_item_box
```

(End of definition for \l__marginalia_item_box.)

`\l__marginalia_item_height_dim` Dimension variables to hold the height and depth of the typeset margin content item.

`\l__marginalia_item_depth_dim`

```
424 \dim_new:N\l__marginalia_item_height_dim
```

```
425 \dim_new:N\l__marginalia_item_depth_dim
```

(End of definition for \l__marginalia_item_height_dim and \l__marginalia_item_depth_dim.)

Variables set by Lua. The following variables will be set by the Lua backend via `tex.count` and `tex.dimen` when `\__marginalia_lua_load_item_data:n` is called.

`\l__marginalia_page_int` Integer variable for the page on which the marginal content item appears. This variable will be made available via `\marginaliapage` within the `\content` of `\marginalia`.

```
426 \int_new:N\l__marginalia_page_int
```

(End of definition for \l__marginalia_page_int.)

`\l__marginalia_column_computed_int` Integer variable for the column next to which the marginal content item appears. This variable will be made available via `\marginaliacolumn` within the `\content` of `\marginalia`.

```
427 \int_new:N\l__marginalia_column_computed_int
```

(End of definition for \l__marginalia_column_computed_int.)

`\l__marginalia_xshift_computed_dim` Dimension variables to hold the differences in x and y coordinates between the call to `\marginalia` and the position where the marginal content item should appear.

`\l__marginalia_yshift_computed_dim`

```
428 \dim_new:N\l__marginalia_xshift_computed_dim
```

```
429 \dim_new:N\l__marginalia_yshift_computed_dim
```

(End of definition for \l__marginalia_xshift_computed_dim and \l__marginalia_yshift_computed_dim.)

`\l__marginalia_side_computed_int` Integer variable to indicate the side of the text block or column on which the marginal content item should be placed: 0 = right and 1 = left.

```
430 \int_new:N\l__marginalia_side_computed_int
```

(This variable could be a boolean, but an integer is used because there is no canonical access to booleans from Lua.)

(End of definition for \l__marginalia_side_computed_int.)

`\l__marginalia_marginno_computed_int` Integer variable to indicate in which margin the content will be placed, to enable quick selection of width and style: 0 = recto outer, 1 = recto inner, 2 = verso outer, 3 = verso inner, 4 = right between, 5 = left between.

```
431 \int_new:N\l__marginalia_marginno_computed_int
```

(End of definition for \l__marginalia_marginno_computed_int.)

`\l__marginalia_enabled_computed_int` Integer variable to indicate whether the marginal content item is enabled: 0 = disabled, 1 = enabled.

```
432 \int_new:N\l__marginalia_enabled_computed_int
```

(This variable could be a boolean, but an integer is used because there is no canonical access to booleans from Lua.)

(End of definition for `\l__marginalia_enabled_computed_int`.)

11.10.2 Core macro

`\__marginalia_process_item:nn` This macro does most of the work in setting the marginal content item. The first parameter is `<options>`, the second is `<content>`.

```
433 \cs_new:Npn\__marginalia_process_item:nn #1#2
434 {
```

First, increment the index, then enter a group where all the action will happen.

```
435 \int_gincr:N\g__marginalia_itemno_int
436 \group_begin:
```

Process `<options>`. These settings apply locally inside the group.

```
437 \keys_set:nn{marginalia}{ #1 }
```

Get item data from the Lua backend: the integer variables `\l__marginalia_page_int`, `\l__marginalia_column_computed_int`, `\l__marginalia_side_computed_int`, `\l__marginalia_enabled_computed_int`, and the dimension variables `\l__marginalia_xshift_computed_dim`, and `\l__marginalia_yshift_computed_dim` are set by Lua via `tex.count` and `tex.dimen`. If no data is available (if, for instance, no data has been stored from a previous run), default values will be set by Lua. On later runs, the Lua backend will supply the values computed from the data written to the `.aux` file on the previous run.

```
438 \__marginalia_lua_load_item_data:n
439 { \int_value:w\g__marginalia_itemno_int }
```

Choose the correct auxiliary function for typesetting, depending on which mode T_EX is in.

```
440 \mode_if_math:TF
441 {
442   \cs_set_eq:NN
443     \__marginalia_typeset:n
444     \__marginalia_typeset_mmode:n
445 }
446 {
447   \legacy_if:nT{@inlabel}
448   { \leavevmode }
449   \mode_if_horizontal:TF
450   {
451     \cs_set_eq:NN
452       \__marginalia_typeset:n
453       \__marginalia_typeset_hmode:n
454   }
455   {
456     \cs_set_eq:NN
457       \__marginalia_typeset:n
458       \__marginalia_typeset_vmode:n
```



```

459         }
460     }

```

Choose the correct box in which to typeset the item for the desired vertical alignment, which has been stored in `\l__marginalia_valign_int`.

```

461     \int_case:nn{\l__marginalia_valign_int}{
462     {
463         {1}{ \cs_set_eq:NN\__marginalia_item_box_set:Nn\ vbox_set_top:Nn }
464         {2}{ \cs_set_eq:NN\__marginalia_item_box_set:Nn\ vbox_set:Nn }
465         {3}{ \cs_set_eq:NN\__marginalia_item_box_set:Nn\ vbox_set_center:Nn }
466         {4}{ \cs_set_eq:NN\__marginalia_item_box_set:Nn\ vbox_set_midway:Nn }
467     }

```

Choose the correct horizontal separation, width, style, and mark for the item.

```

468     \__marginalia_set_xsep_width_style_mark:

```

Typeset the `<content>` into `\l__marginalia_item_box`. Use `\@parboxrestore` for brevity, even though `\hsize` and `\linewidth` are subsequently set to `\l__marginalia_width_dim`. Make available `\marginaliapage` and `\marginaliacolumn`.

```

469     \__marginalia_tagging_socket:n {marginpar/begin}
470     \__marginalia_item_box_set:Nn\l__marginalia_item_box{
471         \@parboxrestore
472         \__marginalia_tagging_socket:n {para/restore}
473         \normalfont\normalsize
474
475         \tl_use:N\l__marginalia_style_tl
476         \dim_set_eq:NN\hsize\l__marginalia_width_dim
477         \dim_set_eq:NN\linewidth\hsize
478
479         \cs_set_eq:NN\marginaliapage\l__marginalia_page_int
480         \cs_set_eq:NN\marginaliacolumn\l__marginalia_column_computed_int
481
482         \group_begin:
483         \ignorespaces
484         #2
485         \par
486         \group_end:
487     }
488     \__marginalia_tagging_socket:n{marginpar/end}

```

Measure `\l__marginalia_item_box`.

```

489     \dim_set:Nn\l__marginalia_item_height_dim
490         {\box_ht:N\l__marginalia_item_box}
491     \dim_set:Nn\l__marginalia_item_depth_dim
492         {\box_dp:N\l__marginalia_item_box}

```

Everything is now ready to place the item on the page and write the necessary data to the `.aux` file. Use the chosen auxiliary function for typesetting, and immediately use `\savepos` to store the callout position.

```

493     \__marginalia_typeset:n{
494         \savepos

```

Write the item data to the `.aux` file. All tokens that will change for future items, and which are currently meaningful, are expanded now; the remainder will be expanded at shipout time, when *they* are meaningful.

```

495         \iow_shipout_e:Ne\l__marginalia_aux_iow{

```

```

496 \token_to_str:N\marginalia@itemdata{
497   itemno=\int_value:w\g__marginalia_itemno_int,
498   abspageno=\exp_not:N\int_eval:n{\g_shipout_readonly_int},
499   pageno=\exp_not:N\int_value:w\c@page,
500   type=\str_use:N\int_value:w\l__marginalia_type_int,
501   xpos=\exp_not:N\int_value:w\lastxpos,
502   ypos=\exp_not:N\int_value:w\lastypos,
503   height=\int_value:w\l__marginalia_item_height_dim,
504   depth=\int_value:w\l__marginalia_item_depth_dim,
505   pos=\int_value:w\l__marginalia_pos_int,
506   column=\int_value:w\l__marginalia_column_int,
507   yshift=\int_value:w\l__marginalia_default_yshift_dim,
508   ysep-above=\int_value:w\l__marginalia_ysep-above_dim,
509   ysep-below=\int_value:w\l__marginalia_ysep-below_dim,
510   ysep~page~top=\int_value:w\l__marginalia_ysep_page_top_dim,
511   ysep~page~bottom=\int_value:w\l__marginalia_ysep_page_bottom_dim,
512 }
513 }

```

Finally, if the item is enabled, typeset it onto the page: shift the item by

$$|\l__marginalia_xshift_computed_dim| + |\l__marginalia_xsep_dim|$$

to the right or left (depending on `\l__marginalia_side_computed_int` in the appropriate overlap `\hbox`, then use `\__marginalia_place_item_box:` for the vertical placement.

```

514 \int_if_zero:nF{\l__marginalia_enabled_computed_int}
515 {
516   \int_if_zero:nTF{\l__marginalia_side_computed_int}
517   {
518     \hbox_overlap_right:n{
519       \kern\l__marginalia_xshift_computed_dim
520       \tl_if_empty:NF\l__marginalia_mark_tl
521       {
522         \hbox_overlap_right:n{ \tl_use:N\l__marginalia_mark_tl }
523       }
524       \kern\l__marginalia_xsep_dim
525       \__marginalia_place_item_box:
526     }
527   }
528   {
529     \hbox_overlap_left:n{
530       \__marginalia_place_item_box:
531       \kern\l__marginalia_xsep_dim
532       \tl_if_empty:NF\l__marginalia_mark_tl
533       {
534         \hbox_overlap_left:n{ \tl_use:N\l__marginalia_mark_tl }
535       }
536       \kern-\l__marginalia_xshift_computed_dim
537     }
538   }
539 }
540 }

```

Close the group started near the beginning of `\__marginalia_process_item:nn`.

```

541 \group_end:
542 }

```

(End of definition for `\_marginolia\_process\_item:nn`.)

11.10.3 Horizontal separation, width, style, mark selection

`\_marginolia\_set\_xsep\_width\_style\_mark:` Set `\l\_marginolia\_xsep\_dim`, `\l\_marginolia\_width\_dim`, and `\l\_marginolia\_style\_tl`, based on `\l\_marginolia\_marginno\_computed\_int`.

```

543 \cs_new:Npn\_marginolia\_set\_xsep\_width\_style\_mark:
544 {
545   \int_case:nn{\l\_marginolia\_marginno\_computed\_int}
546   {
547     {0}
548     {
549       \cs_set_eq:NN\l\_marginolia\_xsep\_dim
550       \l\_marginolia\_xsep\_recto\_outer\_dim
551       \cs_set_eq:NN\l\_marginolia\_width\_dim
552       \l\_marginolia\_width\_recto\_outer\_dim
553       \cs_set_eq:NN\l\_marginolia\_style\_tl
554       \l\_marginolia\_style\_recto\_outer\_tl
555       \cs_set_eq:NN\l\_marginolia\_mark\_tl
556       \l\_marginolia\_mark\_recto\_outer\_tl
557     }
558     {1}
559     {
560       \cs_set_eq:NN\l\_marginolia\_xsep\_dim
561       \l\_marginolia\_xsep\_recto\_inner\_dim
562       \cs_set_eq:NN\l\_marginolia\_width\_dim
563       \l\_marginolia\_width\_recto\_inner\_dim
564       \cs_set_eq:NN\l\_marginolia\_style\_tl
565       \l\_marginolia\_style\_recto\_inner\_tl
566       \cs_set_eq:NN\l\_marginolia\_mark\_tl
567       \l\_marginolia\_mark\_recto\_inner\_tl
568     }
569     {2}
570     {
571       \cs_set_eq:NN\l\_marginolia\_xsep\_dim
572       \l\_marginolia\_xsep\_verso\_outer\_dim
573       \cs_set_eq:NN\l\_marginolia\_width\_dim
574       \l\_marginolia\_width\_verso\_outer\_dim
575       \cs_set_eq:NN\l\_marginolia\_style\_tl
576       \l\_marginolia\_style\_verso\_outer\_tl
577       \cs_set_eq:NN\l\_marginolia\_mark\_tl
578       \l\_marginolia\_mark\_verso\_outer\_tl
579     }
580     {3}
581     {
582       \cs_set_eq:NN\l\_marginolia\_xsep\_dim
583       \l\_marginolia\_xsep\_verso\_inner\_dim
584       \cs_set_eq:NN\l\_marginolia\_width\_dim
585       \l\_marginolia\_width\_verso\_inner\_dim
586       \cs_set_eq:NN\l\_marginolia\_style\_tl
587       \l\_marginolia\_style\_verso\_inner\_tl
588       \cs_set_eq:NN\l\_marginolia\_mark\_tl
589       \l\_marginolia\_mark\_verso\_inner\_tl
590     }
591   }

```

```

591     {4}
592     {
593         \cs_set_eq:NN\l__marginalia_xsep_dim
594         \l__marginalia_xsep_right_between_dim
595         \cs_set_eq:NN\l__marginalia_width_dim
596         \l__marginalia_width_right_between_dim
597         \cs_set_eq:NN\l__marginalia_style_tl
598         \l__marginalia_style_right_between_tl
599         \cs_set_eq:NN\l__marginalia_mark_tl
600         \l__marginalia_mark_right_between_tl
601     }
602     {5}
603     {
604         \cs_set_eq:NN\l__marginalia_xsep_dim
605         \l__marginalia_xsep_left_between_dim
606         \cs_set_eq:NN\l__marginalia_width_dim
607         \l__marginalia_width_left_between_dim
608         \cs_set_eq:NN\l__marginalia_style_tl
609         \l__marginalia_style_left_between_tl
610         \cs_set_eq:NN\l__marginalia_mark_tl
611         \l__marginalia_mark_left_between_tl
612     }
613 }
614 }

```

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

11.10.4 Auxiliary box macros

`\__marginalia_vbox_set_center:Nn` Typesets the second parameter at natural height and stores the result inside a box register supplied as the first parameter, with the baseline being mid-way between the top and bottom of the box.

```

615 \cs_new:Npn \vbox_set_center:Nn #1#2
616 {
617     \vbox_set_top:Nn #1{#2}
618     \dim_set:Nn \l_tmpa_dim{ \box_ht:N#1 + \box_dp:N#1 }
619     \box_set_ht:Nn #1 { .5\l_tmpa_dim }
620     \box_set_dp:Nn #1 { .5\l_tmpa_dim }
621 }

```

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

`\__marginalia_vbox_set_midway:Nn` Typesets the second parameter at natural height and stores the result inside a box register supplied as the first parameter, with the baseline being mid-way between the top and bottom baselines of the items in the box.

```

622 \cs_new:Npn \vbox_set_midway:Nn #1#2
623 {

```

The distance between the topmost and bottommost baselines is equal to the difference in depths between the results of typesetting using a `\vtop` and a `\vbox`. So typeset into both (suspending tagging for the latter), calculate the difference, and adjust the height and depth of the first.

```

624     \vbox_set_top:Nn #1{#2}
625     \tag_suspend:n{ marginalia }

```

```

626 \vbox_set:Nn \l_tmpa_box {#2}
627 \tag_resume:n{ marginalia }
628 \dim_set:Nn \l_tmpa_dim{ \box_dp:N#1 - \box_dp:N\l_tmpa_box}
629 \box_set_ht:Nn #1 { \box_ht:N#1 + .5\l_tmpa_dim }
630 \box_set_dp:Nn #1 { \box_dp:N#1 - .5\l_tmpa_dim }
631 }

```

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

11.10.5 Placement macros

`\__marginalia_place_item_box:` Place the item that has been set in `\l__marginalia_item_box`, vertically shifted by `\l__marginalia_yshift_computed_dim` and `\smashed` to avoid altering vertical spacing in the main text.

```

632 \cs_new:Npn\__marginalia_place_item_box:
633 {
634   \smash
635   {
636     \box_move_up:nn{\l__marginalia_yshift_computed_dim}
637     {
638       \box_use:N\l__marginalia_item_box
639     }
640   }
641 }

```

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

`\__marginalia_typeset_mmode:n` These three macros handle typesetting in math mode, horizontal mode, and vertical mode. Nothing special needs to be done in math mode. In horizontal mode, `\@bsphack...\@esphack` avoids double spacing. In vertical mode, `\if@nobreak` is saved, a new paragraph is started, the item is typeset, the paragraph is ended, a vertical skip of `-\baselineskip` is added, which should ‘hide’ that invisible paragraph, and `\if@nobreak` is restored to the saved value.

```

642 \cs_new:Npn\__marginalia_typeset_mmode:n #1
643 {
644   #1
645 }
646 \cs_new:Npn\__marginalia_typeset_hmode:n #1
647 {
648   \@bsphack
649   #1
650   \@esphack
651 }
652 \bool_new:N\l__marginalia_nobreak_bool
653 \cs_new:Npn\__marginalia_typeset_vmode:n #1
654 {
655   \bool_set:Nn\l__marginalia_nobreak_bool{ \legacy_if_p:n{@nobreak} }
656   \nobreak\noindent #1\par
657   \skip_vertical:n{-\baselineskip}
658   \legacy_if_gset:nn{ @nobreak }{ \l__marginalia_nobreak_bool }
659 }

```

(End of definition for `\__marginalia_typeset_mmode:n`, `\__marginalia_typeset_hmode:n`, and `\__marginalia_typeset_vmode:n`.)

11.11 User commands

Finally, set up the commands for the user.

`\marginalia` This is the main user command for creating a marginal content item. This macro does nothing but hand off to `\__marginalia_process_item:nn`.

```
660 \NewDocumentCommand{\marginalia}{0}{+m }
661 {
662   \__marginalia_process_item:nn{#1}{#2}
663 }
```

(End of definition for \marginalia. This function is documented on page 5.)

`\marginaliasetup` The user command to set the configuration. This macro does nothing but hand off to `\__marginalia_setup:n`.

```
664 \NewDocumentCommand{\marginaliasetup}{m }
665 {
666   \__marginalia_setup:n{ #1 }
667 }
```

(End of definition for \marginaliasetup. This function is documented on page 5.)

`\marginalianewgeometry` The user command to signal that the page geometry has been changed.

```
668 \NewDocumentCommand{\marginalianewgeometry}{}
669 {
670   \__marginalia_write_page_data:
671 }
```

(End of definition for \marginalianewgeometry. This function is documented on page 5.)

```
672 \</package>
```

12 Implementation (Lua backend)

```
673 <lua>
```

12.1 Initial set-up

Module identification/version information.

```
674 local module_name = 'marginalia'
675 local marginalia_module = {
676   name      = module_name,
677   version   = '0.83.30',
678   date      = '2026-07-28',
679   description = 'marginalia',
680   author    = 'Alan J. Cain',
681   copyright  = 'Alan J. Cain',
682   license    = 'LPPL v1.3c'
683 }
684 luatexbase.provides_module(marginalia_module)
```

12.2 Message functions

`info` Functions for writing info or warning messages. First argument is used as a format string
`warning` for later arguments.

```
685 local function info(s,...)
686   luatexbase.module_info(module_name,s:format(...))
687 end
688 local function warning(s,...)
689   luatexbase.module_warning(module_name,s:format(...))
690 end
691 local function error_(s,...)
692   luatexbase.module_error(module_name,s:format(...))
693 end
```

(End of definition for info and warning.)

12.3 Global variables

Global tables for `page_data` and `item_data`.

```
694 local PAGE_DATA_MAIN_TABLE = {}
695 local ITEM_DATA_MAIN_TABLE = {}
```

Global tables for compiling reports.

```
696 local PROBLEM_REPORT_TABLE = {}
697 local PAGE_CHANGE_REPORT_TABLE = {}
698 local ITEM_CHANGE_REPORT_TABLE = {}
```

12.4 Constants

Type constants. These match the possible values for the `type` key.

```
699 local TYPE_NORMAL = 1
700 local TYPE_FIXED = 2
701 local TYPE_OPTFIXED = 3
```

Position constants. These match the possible values for the `pos` key.

```
702 local POS_AUTO = 1
703 local POS_REVERSE = 2
704 local POS_LEFT = 3
705 local POS_RIGHT = 4
706 local POS_NEAREST = 5
```

12.5 Keys for tables

The strings listed in this subsection are constants used to index the tables. Also listed are the types of values that are indexed by each key. Note that values listed below as **dimensions** are actually integers, giving the dimension in $\text{\TeX}$ scaled points (sp)

12.5.1 Keys for both page and item data tables

Integer: Absolute page number in output file (not on-page number), used in both `page_data` and `item_data` tables

```
707 local KEY_ABSPAGENO = 'abspageno'
```

Boolean: Used to mark `page_data` or `item_data` as checked when the `.aux` file is read back at the end of the document

```
708 local KEY_CHECKED = 'checked'
```

12.5.2 Keys for page data tables, layout etc.

Integer: Used only to distinguish instances of data written to `.aux` file

```
709 local KEY_PAGEDATANO = 'pagedatano'
```

Dimensions: Value of next two will always be equivalent of 1 in, but it is simpler to keep all geometry data together.

```
710 local KEY_HOFFSETORIGIN = 'hoffsetorigin'
```

```
711 local KEY_VOFFSETORIGIN = 'voffsetorigin'
```

Dimensions: corresponding to obvious $\text{\LaTeX}$ dimensions

```
712 local KEY_HOFFSET = 'hoffset'
```

```
713 local KEY_VOFFSET = 'voffset'
```

```
714 local KEY_PAGEHEIGHT = 'pageheight'
```

```
715 local KEY_ODDSIDEMARGIN = 'oddsidemargin'
```

```
716 local KEY_EVENSIDEMARGIN = 'evensidemargin'
```

```
717 local KEY_TEXTWIDTH = 'textwidth'
```

```
718 local KEY_COLUMNWIDTH = 'columnwidth'
```

```
719 local KEY_COLUMNSEP = 'columnsep'
```

Integer: either 1 or 2, depending on whether $\text{\LaTeX}$ was in one- or two-column mode

```
720 local KEY_COLUMNCOUNT = 'columncount'
```

Boolean: true iff $\text{\LaTeX}$ is in twoside mode

```
721 local KEY_TWOSIDE = 'twoside'
```

12.5.3 Keys for item data tables

Integer: Used to identify data with item

```
722 local KEY_ITEMNO = 'itemno'
```

Integer: On-page number

```
723 local KEY_PAGENO = 'pageno'
```

Dimensions: x and y positions of call to `\marginalia`

```
724 local KEY_XPOS = 'xpos'
```

```
725 local KEY_YPOS = 'ypos'
```

Dimensions: Height and depth of typeset item

```
726 local KEY_HEIGHT = 'height'
```

```
727 local KEY_DEPTH = 'depth'
```

Integer: Specified type, following `TYPE_*`

```
728 local KEY_TYPE = 'type'
```


Integer: corresponds to value of `pos` key: 0 = auto, 1 = reverse, 2 = left, 3 = right, 4 = nearest

```
729 local KEY_POS = 'pos'
```

Integer: corresponds to value of `column` key: -1 = auto, 0 = one, 1 = left, 2 = right

```
730 local KEY_COLUMN = 'column'
```

Dimension: specified vertical shift

```
731 local KEY_YSHIFT = 'yshift'
```

Dimensions: specified vertical separations

```
732 local KEY_YSEP_ABOVE = 'ysep above'
```

```
733 local KEY_YSEP_BELOW = 'ysep below'
```

```
734 local KEY_YSEP_PAGE_TOP = 'ysep page top'
```

```
735 local KEY_YSEP_PAGE_BOTTOM = 'ysep page bottom'
```

The preceding keys refer to values that will be supplied from L^AT_EX. The remaining values will be computed in Lua and passed back to L^AT_EX.

Integer: column in which the call to `\marginalia` was located: 0 = one-column, 1 = left, 2 = right

```
736 local KEY_COLNO_COMPUTED = 'colno computed'
```

Dimension: Horizontal shift between the call to `\marginalia` and the margin in which the item should be located

```
737 local KEY_XSHIFT_COMPUTED = 'xshift computed'
```

Dimension: Computed vertical shift

```
738 local KEY_YSHIFT_COMPUTED = 'yshift computed'
```

Integer: Side of text on which the item will appear: 0 = right, 1 = left

```
739 local KEY_SIDE_COMPUTED = 'side computed'
```

Integer: Number of margin in which the item will appear, 0 = recto outer, 1 = recto inner, 2 = verso outer, 3 = verso inner, 4 = right between, 5 = left between

```
740 local KEY_MARGINNO_COMPUTED = 'marginno computed'
```

Boolean: Whether the item will actually appear on the page

```
741 local KEY_ENABLED_COMPUTED = 'enabled computed'
```

12.6 Utility functions

`list_filter`

12 Code adapted from

<https://stackoverflow.com/a/53038524>.

Take a list `t` and remove from it any elements for which the function `f` does not return true. (The index `j` is always the destination index to which a ‘keep’ element is moved.)¹²

```
742 local function list_filter(t, f)
```

```
743   local j = 1
```

```
744   local n = #t
```

```
745
```

```
746   for i=1,n do
```

```
747     if (f(t[i])) then
```

```
748       if (i ~= j) then
```

```
749         t[j] = t[i]
```

```
750         t[i] = nil
```

```
751       end
```

```
752       j = j + 1
```

```
753     else
```

```

754         t[i] = nil
755     end
756 end
757
758 end

```

(End of definition for list_filter.)

list_filter Return boolean true iff *s* is exactly the string ‘true’.

```

759 local function toboolean(s)
760     return s == "true"
761 end

```

(End of definition for list_filter.)

get_data_page_number Take a item or page data and return a human-readable string indicating the page to which the data pertains.

```

762 local function get_data_page_number(data)
763     local pageno = data[KEY_PAGENO]
764     if pageno ~= nil then
765         return 'p' .. pageno .. ' (' .. data[KEY_ABSPAGENO] .. ')'
766     else
767         return data[KEY_ABSPAGENO]
768     end
769 end

```

(End of definition for get_data_page_number.)

12.7 Generic page/item data functions

parse_data Parse *keyvalue_string* and return the corresponding data as a table. The *keyvalue_string* is expected to be of precisely the kind written to the *.aux* file as the parameter of *\marginalia@pagedata* or *\marginalia@notedata*.

Ignore any keys in *keyvalue_string* that are not listed in *conversion_table*. Fill in any missing value with values from *defaults_table*.

conversion_table is indexed by possible keys, with values equal to functions to convert the corresponding value string to the value that should appear in the returned table.

defaults_table is indexed by keys that *will* appear in the returned table, using the corresponding value unless it was given in *keyvalue_string* and the key appeared in *conversion_table*.

```

770 local function parse_data(keyvalue_string,conversion_table,defaults_table)
771
772     local key
773     local value
774     local result = {}
775
776     for s in string.gmatch(keyvalue_string,'([^\,]+)') do
777
778         key,value = string.match(s,'^(.+)=(.+)$')
779         local conv = conversion_table[key]
780         if conv ~= nil then
781             result[key] = conv(value)

```

```

782     end
783
784 end
785
786 for key,value in pairs(defaults_table) do
787     if not(result[key] ~= nil) then
788         result[key] = value
789     end
790 end
791
792 return result
793
794 end

```

(End of definition for parse_data.)

check_data Check keyvalue_string against stored data. If it is new or has changed, append a report to report_table. Set the KEY_CHECKED of the data item to true.

The keyvalue_string is processed using conversion_table and defaults_table as per the parse_data function. The resulting table is compared to the table in data_table with the same value whose key is data_table_key. The tables are compared using the fields indexed by keys in conversion_table.

```

795 local function check_data(keyvalue_string,conversion_table,defaults_table,
796                           data_table,data_table_key_field,report_table)
797
798     local new_data = parse_data(keyvalue_string,
799                                 conversion_table,defaults_table)
800
801     local data_table_key = new_data[data_table_key_field]
802
803     local stored_data = data_table[data_table_key]
804     if stored_data == nil then
805         table.insert(
806             report_table,
807             get_data_page_number(new_data) .. ' New'
808         )
809     else
810         local change_report = ''
811         for k,_ in pairs(conversion_table) do
812             if stored_data[k] ~= new_data[k] then
813                 change_report = change_report
814                     .. ' ' .. k .. ':' ..
815                     tostring(stored_data[k]) .. '->' .. tostring(new_data[k])
816             end
817         end
818         if change_report ~= '' then
819             table.insert(
820                 report_table,
821                 get_data_page_number(new_data) .. ' ' .. change_report
822             )
823         end
824         stored_data[KEY_CHECKED] = true
825     end
826

```

827 end

(End of definition for check_data.)

check_removed_data Check whether data have been removed from data_table, which corresponds to some entry having the value of KEY_CHECKED being false. In this case, append a report to report_table.

```

828 local function check_removed_data(data_table,report_table)
829   for _,data in pairs(data_table) do
830     if not data[KEY_CHECKED] then
831       table.insert(
832         report_table,
833         ' Removed'
834       )
835       break
836     end
837   end
838 end

```

(End of definition for check_removed_data.)

12.8 Processing of page data from .aux file

Conversion and default tables.

```

839 local PAGE_DATA_CONVERSION_TABLE = {
840   [KEY_PAGEDATANO] = tonumber,
841   [KEY_ABSPAGENO] = tonumber,
842   [KEY_HOFFSETORIGIN] = tonumber,
843   [KEY_VOFFSETORIGIN] = tonumber,
844   [KEY_HOFFSET] = tonumber,
845   [KEY_VOFFSET] = tonumber,
846   [KEY_PAGEHEIGHT] = tonumber,
847   [KEY_ODDSIDEMARGIN] = tonumber,
848   [KEY_EVENSIDEMARGIN] = tonumber,
849   [KEY_COLUMNCOUNT] = tonumber,
850   [KEY_COLUMNWIDTH] = tonumber,
851   [KEY_COLUMNSEP] = tonumber,
852   [KEY_TEXTWIDTH] = tonumber,
853   [KEY_TWOSIDE] = toboolean,
854 }
855 local PAGE_DATA_DEFAULT_TABLE = {
856   [KEY_PAGEDATANO] = 0,
857   [KEY_ABSPAGENO] = 0,
858   [KEY_HOFFSETORIGIN] = tex.sp('1in'),
859   [KEY_VOFFSETORIGIN] = tex.sp('1in'),
860   [KEY_HOFFSET] = tex.dimen['hoffset'],
861   [KEY_VOFFSET] = tex.dimen['voffset'],
862   [KEY_PAGEHEIGHT] = tex.dimen['pageheight'],
863   [KEY_ODDSIDEMARGIN] = tex.dimen['oddsidemargin'],
864   [KEY_EVENSIDEMARGIN] = tex.dimen['evensidemargin'],
865   [KEY_TEXTWIDTH] = tex.dimen['textwidth'],
866   [KEY_COLUMNWIDTH] = tex.dimen['columnwidth'],
867   [KEY_COLUMNSEP] = tex.dimen['columnsep'],
868   [KEY_COLUMNCOUNT] = 1,

```

```

869     [KEY_TWOSIDE] = false,
870     [KEY_CHECKED] = false,
871 }

```

store_page_data Store page data supplied by keyvalue_string in PAGE_DATA_MAIN_TABLE.

```

872 local function store_page_data(keyvalue_string)
873
874     local page_data = parse_data(keyvalue_string,
875                                 PAGE_DATA_CONVERSION_TABLE,
876                                 PAGE_DATA_DEFAULT_TABLE)
877
878     PAGE_DATA_MAIN_TABLE[page_data[KEY_PAGEDATANO]] = page_data
879
880 end

```

(End of definition for store_page_data.)

store_default_page_data Store default page data in PAGE_DATA_MAIN_TABLE, so that there is some data to work with when computing item positions, even on a first run, when no page data has been written to the .aux file.

```

881 local function store_default_page_data()
882
883     default_page_data = parse_data('',
884                                     PAGE_DATA_CONVERSION_TABLE,
885                                     PAGE_DATA_DEFAULT_TABLE)
886
887     default_page_data[KEY_ABSPAGENO] = 1
888     default_page_data[KEY_CHECKED] = true
889
890     PAGE_DATA_MAIN_TABLE[0] = default_page_data
891
892 end

```

(End of definition for store_default_page_data.)

check_page_data Check whether page_data supplied by keyvalue_string differs from that in PAGE_DATA_MAIN_TABLE, appending reports to PAGE_CHANGE_REPORT_TABLE if so.

```

893 local function check_page_data(keyvalue_string)
894
895     check_data(keyvalue_string,
896                PAGE_DATA_CONVERSION_TABLE, PAGE_DATA_DEFAULT_TABLE,
897                PAGE_DATA_MAIN_TABLE, KEY_PAGEDATANO,
898                PAGE_CHANGE_REPORT_TABLE)
899
900 end

```

(End of definition for check_page_data.)

12.9 Processing of item data from .aux file

Conversion and default tables.

```

901 local ITEM_DATA_CONVERSIONS = {
902     [KEY_ITEMNO] = tonumber,
903     [KEY_ABSPAGENO] = tonumber,

```

```

904 [KEY_PAGENO] = tonumber,
905 [KEY_XPOS] = tonumber,
906 [KEY_YPOS] = tonumber,
907 [KEY_HEIGHT] = tonumber,
908 [KEY_DEPTH] = tonumber,
909 [KEY_TYPE] = tonumber,
910 [KEY_POS] = tonumber,
911 [KEY_COLUMN] = tonumber,
912 [KEY_YSHIFT] = tonumber,
913 [KEY_YSEP_ABOVE] = tonumber,
914 [KEY_YSEP_BELOW] = tonumber,
915 [KEY_YSEP_PAGE_TOP] = tonumber,
916 [KEY_YSEP_PAGE_BOTTOM] = tonumber,
917 [KEY_CHECKED] = toboolean,
918 }
919 local ITEM_DATA_DEFAULTS = {
920 [KEY_ITEMNO] = 0,
921 [KEY_ABSPAGENO] = 1,
922 [KEY_PAGENO] = 1,
923 [KEY_XPOS] = 0,
924 [KEY_YPOS] = 0,
925 [KEY_HEIGHT] = 0,
926 [KEY_DEPTH] = 0,
927 [KEY_TYPE] = 0,
928 [KEY_POS] = 0,
929 [KEY_COLUMN] = -1,
930 [KEY_YSHIFT] = 0,
931 [KEY_YSEP_ABOVE] = tex.dimen['marginparpush'],
932 [KEY_YSEP_BELOW] = tex.dimen['marginparpush'],
933 [KEY_YSEP_PAGE_TOP] = tex.dimen['marginparpush'],
934 [KEY_YSEP_PAGE_BOTTOM] = tex.dimen['marginparpush'],
935 [KEY_COLNO_COMPUTED] = 0,
936 [KEY_XSHIFT_COMPUTED] = 0,
937 [KEY_YSHIFT_COMPUTED] = 0,
938 [KEY_SIDE_COMPUTED] = 0,
939 [KEY_MARGINNO_COMPUTED] = 0,
940 [KEY_ENABLED_COMPUTED] = true,
941 [KEY_CHECKED] = false,
942 }

```

ITEM_DATA_DEFAULTS is also used by load_item_data when no stored item data is found in ITEM_DATA_MAIN_TABLE.

store_item_data Store item_data supplied by keyvalue_string in ITEM_DATA_MAIN_TABLE.

```

943 local function store_item_data(keyvalue_string)
944
945     local item = parse_data(keyvalue_string,
946                             ITEM_DATA_CONVERSIONS,
947                             ITEM_DATA_DEFAULTS)
948
949     ITEM_DATA_MAIN_TABLE[item[KEY_ITEMNO]] = item
950
951 end

```

(End of definition for store_item_data.)

check_item_data Check whether item_data supplied by keyvalue_string differs from that in ITEM_DATA_MAIN_TABLE, appending reports to ITEM_CHANGE_REPORT_TABLE if so.

```

952 local function check_item_data(keyvalue_string)
953
954     check_data(keyvalue_string,
955                 ITEM_DATA_CONVERSIONS, ITEM_DATA_DEFAULTS,
956                 ITEM_DATA_MAIN_TABLE, KEY_ITEMNO,
957                 ITEM_CHANGE_REPORT_TABLE)
958
959 end

```

(End of definition for check_item_data.)

12.10 Writing reports

write_report If report_table is non-empty, write the data contained in it as a module warning. Use at most max_length items. text is written before the report. If max_length is non-zero, after_items_text is written after the items. noun refers to the kind of items.

```

960 local function write_report(report_table,max_length,text,after_items_text,noun)
961
962     if #report_table == 0 then
963         return
964     end
965
966     if max_length == 0 then
967         warning(text)
968         return
969     end
970
971     local report_text
972     local report_length
973
974     if #report_table <= max_length then
975         report_length = #report_table
976         report_text = ' Here are the ' .. noun .. ':\n'
977     else
978         report_length = max_length
979         report_text = ' Here are the first ' .. report_length .. ' ' .. noun .. ':\n'
980     end
981
982     for i=1,report_length do
983         report_text = report_text .. report_table[i]
984         if i < report_length then
985             report_text = report_text .. '\n'
986         end
987     end
988
989     warning(text .. ' ' .. report_text .. '\n' .. after_items_text)
990
991 end

```

(End of definition for write_report.)

write_problem_report Write a report about placement problems to T_EX in a format suitable for a package warning.

```

992 local function write_problem_report()
993
994   write_report(
995     PROBLEM_REPORT_TABLE,
996     tex.count['l__marginalia_max_problem_int'],
997     'Problems in placement.',
998     'Problems listed at \\end{document}',
999     'problems'
1000   )
1001
1002
1003 end

```

(End of definition for write_problem_report.)

write_item_change_report Write a report about changes in item data to T_EX in a format suitable for a package warning.

```

1004 local function write_item_change_report()
1005
1006   check_removed_data(ITEM_DATA_MAIN_TABLE, ITEM_CHANGE_REPORT_TABLE)
1007   write_report(
1008     ITEM_CHANGE_REPORT_TABLE,
1009     tex.count['l__marginalia_max_item_data_change_int'],
1010     'Changes in item data. Rerun to get correct placement.',
1011     'Changes listed at \\end{document}',
1012     'changes'
1013   )
1014
1015 end

```

(End of definition for write_item_change_report.)

write_page_change_report Write a report about changes in page data to T_EX in a format suitable for a package warning.

```

1016 local function write_page_change_report()
1017
1018   check_removed_data(PAGE_DATA_MAIN_TABLE, PAGE_CHANGE_REPORT_TABLE)
1019   write_report(
1020     PAGE_CHANGE_REPORT_TABLE,
1021     tex.count['l__marginalia_max_page_data_change_int'],
1022     'Changes in page data. Rerun to get correct placement.',
1023     'Changes listed at \\end{document}',
1024     'changes'
1025   )
1026
1027 end

```

(End of definition for write_page_change_report.)

12.11 Computing horizontal positions

It is necessary to determine whether an item should be placed on the right or left of the text block, and in which column it lies. The following lookup tables are used.

The value found in `RIGHTSIDE_LOOKUP_TABLE` is either `true` (right) or `false` (left). It is indexed by whether the item is on a recto page (`true/false`), whether it pertains to single-column text, the left column, or the right column (0/1/2), and the value of `pos` being either `auto` or `reverse`.

```
1028 local RIGHTSIDE_LOOKUP_TABLE = {
1029   [true] = {
1030     [0] = {
1031       [POS_AUTO] = true,
1032       [POS_REVERSE] = false,
1033     },
1034     [1] = {
1035       [POS_AUTO] = false,
1036       [POS_REVERSE] = true,
1037     },
1038     [2] = {
1039       [POS_AUTO] = true,
1040       [POS_REVERSE] = false,
1041     },
1042   },
1043   [false] = {
1044     [0] = {
1045       [POS_AUTO] = false,
1046       [POS_REVERSE] = true,
1047     },
1048     [1] = {
1049       [POS_AUTO] = false,
1050       [POS_REVERSE] = true,
1051     },
1052     [2] = {
1053       [POS_AUTO] = true,
1054       [POS_REVERSE] = false,
1055     },
1056   },
1057 }
```

The value found in `MARGINNO_LOOKUP_TABLE` ranges from 0 to 5 (see `KEY_MARGINNO_COMPUTED` for the meaning of these values). It is indexed by whether the item is on a recto page (`true/false`), whether it pertains to single-column text, the left column, or the right column (0/1/2), and whether it is to be placed on the right of the text block (`true/false`).

```
1058 local MARGINNO_LOOKUP_TABLE = {
1059   [true] = {
1060     [0] = {
1061       [false] = 1,
1062       [true] = 0,
1063     },
1064     [1] = {
1065       [false] = 1,
1066       [true] = 5,
1067     },
1068   },
1069 }
```

```

1068     [2] = {
1069         [false] = 4,
1070         [true] = 0,
1071     },
1072 },
1073 [false] = {
1074     [0] = {
1075         [false] = 2,
1076         [true] = 3,
1077     },
1078     [1] = {
1079         [false] = 2,
1080         [true] = 5,
1081     },
1082     [2] = {
1083         [false] = 4,
1084         [true] = 3,
1085     },
1086 },
1087 }

```

`compute_items_horizontal` For every `item_data` in `item_data_list`, compute the fields relevant to horizontal positioning, namely `KEY_COLNO_COMPUTED`, `KEY_XSHIFT_COMPUTED`, `KEY_SIDE_COMPUTED`, based on the layout information in `page_data`. Every item described in `item_data_list` is assumed to be on the same page.

```

1088 local function compute_items_horizontal(item_data_list,page_data)
Immediately return if item_data_list is empty, to avoid edge cases.
1089     if #item_data_list == 0 then
1090         return
1091     end

```

Information used frequently and which is the same for every item.

```

1092     local pageno = item_data_list[1][KEY_PAGENO]
1093     local twoside = page_data[KEY_TWOSIDE]
1094     local recto = ((pageno % 2) == 1) or (not twoside)
1095     local columncount = page_data[KEY_COLUMNCOUNT]

```

Tables to contain the x -coordinates of left edge, right edge, and middle of the current text, whether a single column (index 0), the left column (index 1), or the right column (index 2).

```

1096     local x_textleft = {}
1097     local x_textright = {}
1098     local x_textmiddle = {}

```

First, compute necessary dimensions for single-column text, since most of these calculations would be used anyway for two-column text. The terms used in calculating `x_textleft[0]` respectively take one to the origin of `\hoffset`, to the origin of `\oddsidemargin` and `\evensidemargin`, and to the left-hand side of the text block.

```

1099     if recto then
1100         x_textleft[0] = (
1101             page_data[KEY_HOFFSETORIGIN]
1102             + page_data[KEY_HOFFSET]
1103             + page_data[KEY_ODDSIDEMARGIN]
1104         )

```

```

1105     x_textright[0] = (
1106         x_textleft[0]
1107         + page_data[KEY_TEXTWIDTH]
1108     )
1109 else
1110     x_textleft[0] = (
1111         page_data[KEY_HOFFSETORIGIN]
1112         + page_data[KEY_HOFFSET]
1113         + page_data[KEY_EVENSIDEMARGIN]
1114     )
1115     x_textright[0] = (
1116         x_textleft[0]
1117         + page_data[KEY_TEXTWIDTH]
1118     )
1119 end
1120 x_textmiddle[0] = (x_textleft[0] + x_textright[0])/2
1121
1122
1123 if columncount == 1 then

```

If the page is one-column, the field KEY_COLNO_COMPUTED can be set immediately for every item_data.

```

1124     for i=1,#item_data_list do
1125         item_data_list[i][KEY_COLNO_COMPUTED] = 0
1126     end
1127 else

```

If the page is two-column, calculate the *x*-coordinates of the left and right edges and the mid-point of each column.

```

1128     x_textleft[1] = x_textleft[0]
1129     x_textright[1] = (
1130         x_textleft[1]
1131         + page_data[KEY_COLUMNWIDTH]
1132     )
1133     x_textmiddle[1] = (x_textleft[1] + x_textright[1])/2
1134
1135     x_textleft[2] = (
1136         x_textright[1]
1137         + page_data[KEY_COLUMNSEP]
1138     )
1139     x_textright[2] = (
1140         x_textleft[2]
1141         + page_data[KEY_COLUMNWIDTH]
1142     )
1143     x_textmiddle[2] = (x_textleft[2] + x_textright[2])/2
1144

```

Calculate the cut-off (mid-way between the columns) that distinguishes items from left and right columns.

```

1145     local left_column_x_limit = (
1146         x_textright[1]
1147         + .5*page_data[KEY_COLUMNSEP]
1148     )

```

Now set the field KEY_COLNO_COMPUTED for each item.

```

1149   for i=1,#item_data_list do
1150     local item_data = item_data_list[i]
1151
1152     if item_data[KEY_COLUMN] >= 0 then
1153       item_data[KEY_COLNO_COMPUTED] = item_data[KEY_COLUMN]
1154     else
1155       if item_data[KEY_XPOS] <= left_column_x_limit then
1156         item_data[KEY_COLNO_COMPUTED] = 1
1157       else
1158         item_data[KEY_COLNO_COMPUTED] = 2
1159       end
1160     end
1161   end
1162 end
1163 end

```

For every item_data in item_data_list, compute and set the fields KEY_SIDE_COMPUTED, KEY_XSHIFT_COMPUTED, and KEY_MARGINNO_COMPUTED.

```

1164   for i=1,#item_data_list do
1165     local item = item_data_list[i]
1166
1167     local pos = item[KEY_POS]
1168     local colnocomputed = item[KEY_COLNO_COMPUTED]
1169
1170     if pos == POS_LEFT then
1171       rightside = false
1172     elseif pos == POS_RIGHT then
1173       rightside = true
1174     elseif pos == POS_NEAREST then
1175       rightside = (item[KEY_XPOS] >= x_textmiddle[colnocomputed])
1176     else

```

(In this case, pos must be POS_AUTO or POS_REVERSE.)

```

1177       rightside = RIGHTSIDE_LOOKUP_TABLE[recto][colnocomputed][pos]
1178     end
1179
1180     local marginno = MARGINNO_LOOKUP_TABLE[recto][colnocomputed][rightside]
1181
1182     if rightside then
1183       item[KEY_SIDE_COMPUTED] = 0
1184       item[KEY_XSHIFT_COMPUTED] = -item[KEY_XPOS]
1185                                   + x_textright[colnocomputed]
1186     else
1187       item[KEY_SIDE_COMPUTED] = 1
1188       item[KEY_XSHIFT_COMPUTED] = -item[KEY_XPOS]
1189                                   + x_textleft[colnocomputed]
1190     end
1191     item[KEY_MARGINNO_COMPUTED] = marginno
1192
1193   end
1194 end
1195 end

```

(End of definition for compute_items_horizontal.)

12.12 Computing vertical positions

12.12.1 Auxiliary functions

`get_y_item_top` Return the y -coordinate of the top of the item described by `item_data`.

```
1196 local function get_y_item_top(item_data)
1197   return item_data[KEY_YPOS]
1198         + item_data[KEY_YSHIFT_COMPUTED]
1199         + item_data[KEY_HEIGHT]
1200 end
```

(End of definition for get_y_item_top.)

`get_y_item_bottom` Return the y -coordinate of the bottom of the item described by `item_data`.

```
1201 local function get_y_item_bottom(item_data)
1202   return item_data[KEY_YPOS]
1203         - item_data[KEY_DEPTH]
1204         + item_data[KEY_YSHIFT_COMPUTED]
1205 end
```

(End of definition for get_y_item_bottom.)

`get_ysep_list` Calculate the separation to be used between adjacent marginal content items as described in `item_data_list`. The list is assumed to be sorted so that items are in the order they should appear on the page, top to bottom.

The idea is that we have the following arrangement for $i = 1, \dots, \#item_data_list$:

```
⋮
item_data_list[i]
ysep_list[i]
item_data_list[i+1]
⋮
```

Also set `ysep_list[0]` and `ysep_list[#item_data_list]` to 0, to avoid checking when these values are accessed (although they are not used).

```
1206 local function get_ysep_list(item_data_list)
1207
1208   local ysep_list = {}
1209
1210   ysep_list[0] = 0
1211   for i=1,#item_data_list-1 do
1212     ysep_list[i] = math.max(
1213                           item_data_list[i][KEY_YSEP_BELOW],
1214                           item_data_list[i+1][KEY_YSEP_ABOVE]
1215                        )
1216   end
1217   ysep_list[#item_data_list] = 0
1218
1219   return ysep_list
1220
1221 end
```

(End of definition for get_ysep_list.)

12.12.2 Computing optfixed enabled

compute_items_vertical_optfixed_enabled

For every `item_data` in `item_data_list` describing an item of type `TYPE_OPTFIXED`, check for a clash with an item of type `TYPE_FIXED`. If so, set `item_data[KEY_ENABLED_COMPUTED]` to false. Every item described in `item_data_list` is assumed to be on the same page and to have `KEY_YSHIFT` set to the default.

```
1222 local function compute_items_vertical_optfixed_enabled(item_data_list)
1223
1224   local optfixed_item_data_list = {}
1225   local fixed_item_data_list = {}
1226
1227   for _,item_data in pairs(item_data_list) do
1228     if item_data[KEY_TYPE] == TYPE_OPTFIXED then
1229       optfixed_item_data_list[#optfixed_item_data_list+1] = item_data
1230     elseif item_data[KEY_TYPE] == TYPE_FIXED then
1231       fixed_item_data_list[#fixed_item_data_list+1] = item_data
1232     end
1233   end
1234
1235   for _,optfixed_item_data in pairs(optfixed_item_data_list) do
1236     local optfixed_y_item_top = get_y_item_top(optfixed_item_data)
1237     local optfixed_y_item_bottom = get_y_item_bottom(optfixed_item_data)
1238
1239     for _,fixed_item_data in pairs(fixed_item_data_list) do
1240       local fixed_y_item_top = get_y_item_top(fixed_item_data)
1241       local fixed_y_item_bottom = get_y_item_bottom(fixed_item_data)
1242
1243       if (
1244         (
1245           (fixed_y_item_bottom - optfixed_y_item_top)
1246           <
1247           math.max(
1248             fixed_item_data[KEY_YSEP_BELOW],
1249             optfixed_item_data[KEY_YSEP_ABOVE]
1250           )
1251         )
1252         and
1253         (
1254           (optfixed_y_item_bottom - fixed_y_item_top)
1255           <
1256           math.max(
1257             optfixed_item_data[KEY_YSEP_BELOW],
1258             fixed_item_data[KEY_YSEP_ABOVE]
1259           )
1260         )
1261       ) then
1262         optfixed_item_data[KEY_ENABLED_COMPUTED] = false
1263         break
1264       end
1265     end
1266   end
1267
1268 end
```

(End of definition for compute_items_vertical_optfixed_enabled.)

12.12.3 Computing vertical adjustment

compute_items_vertical_adjustment

For every `item_data` in `item_data_list`, compute the field relevant to vertical positioning, namely `KEY_YSHIFT_COMPUTED`, based on the layout information in `page_data`. Every item described in `item_data_list` is assumed to be on the same page and to have `KEY_YSHIFT` set to the default, and the list is assumed to be sorted so that items are in the order they should appear on the page, top to bottom.

```
1269 local function compute_items_vertical_adjustment(item_data_list,page_data)
```

Immediately return if `item_data_list` is empty, to avoid edge cases

```
1270 if #item_data_list == 0 then
1271     return
1272 end
```

```
1273
1274 local ysep_list = get_ysep_list(item_data_list)
```

First pass of computation (downward). `y_limit_above` will always be the highest *y*-coordinate at which the top of next item below can appear.

```
1275 local y_limit_above = (
1276     page_data[KEY_VOFFSET]
1277     + page_data[KEY_PAGEHEIGHT]
1278     - item_data_list[1][KEY_YSEP_PAGE_TOP]
1279 )
1280
1281 for i=1,#item_data_list do
1282     local item_data = item_data_list[i]
1283
1284     local y_item_top = get_y_item_top(item_data)
1285
1286     if y_item_top > y_limit_above then
1287         if item_data[KEY_TYPE] == TYPE_NORMAL then
1288             item_data[KEY_YSHIFT_COMPUTED] = item_data[KEY_YSHIFT_COMPUTED]
1289                                             + (y_limit_above - y_item_top)
1290         end
1291     end
1292
1293     y_limit_above = get_y_item_bottom(item_data) - ysep_list[i]
1294 end
```

Second pass of computation (upward). `y_limit_below` will always be the lowest *y*-coordinate at which the bottom of next item above can appear.

```
1295 local y_limit_below = (
1296     page_data[KEY_VOFFSET]
1297     + item_data_list[#item_data_list][KEY_YSEP_PAGE_BOTTOM]
1298 )
1299
1300 for i=#item_data_list,1,-1 do
1301     local item_data = item_data_list[i]
1302
1303     local y_item_bottom = get_y_item_bottom(item_data)
1304
1305     if y_item_bottom < y_limit_below then
```

```

1306         if item_data[KEY_TYPE] == TYPE_NORMAL then
1307             item_data[KEY_YSHIFT_COMPUTED] = item_data[KEY_YSHIFT_COMPUTED]
1308                                             + (y_limit_below - y_item_bottom)
1309         end
1310     end
1311
1312     y_limit_below = get_y_item_top(item_data) + ysep_list[i-1]
1313 end
1314
1315 end

```

(End of definition for compute_items_vertical_adjustment.)

12.12.4 Checking vertical adjustment

Messages to use when checking results of vertical adjustment.

```

1316 local ITEM_PASSED_YSEP_PAGE_TOP_MESSAGES = {
1317     [TYPE_NORMAL] = 'Moveable item > ysep page top',
1318     [TYPE_FIXED] = 'Topmost fixed item > ysep page top',
1319     [TYPE_OPTFIXED] = 'Topmost optfixed item > ysep page top',
1320 }
1321 local ITEM_CLASH_MESSAGES = {
1322     [TYPE_NORMAL] = {
1323         [TYPE_NORMAL] = 'moveable items'
1324                     .. ' (this shouldn\'t happen)',
1325         [TYPE_FIXED] = 'moveable item above fixed item',
1326         [TYPE_OPTFIXED] = 'moveable item above optfixed item',
1327     },
1328     [TYPE_FIXED] = {
1329         [TYPE_NORMAL] = 'moveable item below fixed item',
1330         [TYPE_FIXED] = 'fixed items',
1331         [TYPE_OPTFIXED] = 'fixed item above optfixed item '
1332                     .. ' (this shouldn\'t happen)',
1333     },
1334     [TYPE_OPTFIXED] = {
1335         [TYPE_NORMAL] = 'moveable items below optfixed item',
1336         [TYPE_FIXED] = 'fixed item below optfixed item '
1337                     .. ' (this shouldn\'t happen)',
1338         [TYPE_OPTFIXED] = 'optfixed items '
1339                     .. ' (this shouldn\'t happen)',
1340     },
1341 }
1342 local ITEM_PASSED_YSEP_PAGE_BOTTOM_MESSAGE = {
1343     [TYPE_NORMAL] = 'Moveable item < ysep page bottom',
1344     [TYPE_FIXED] = 'Bottommost fixed item < ysep page bottom',
1345     [TYPE_OPTFIXED] = 'Bottommost optfixed item < ysep page bottom',
1346 }

```

`check_items_vertical` For the items described by the `item_data` in `item_data_list`, check whether any clash or fail to obey ysep page top or ysep page bottom. If so, write messages to `PROBLEM_REPORT_TABLE`.

```

1347 local function check_items_vertical(item_data_list,page_data)

```


Immediately return if item_data_list is empty, to avoid edge cases

```
1348   if (#item_data_list) == 0 then
1349       return
1350   end
1351
1352   local ysep_list = get_ysep_list(item_data_list)
1353
1354   local item_data
1355
```

If any item fails to obey ysep page top, the first one in the list does.

```
1356   item_data = item_data_list[1]
1357   if (
1358       get_y_item_top(item_data) > page_data[KEY_VOFFSET]
1359                               + page_data[KEY_PAGEHEIGHT]
1360                               - item_data[KEY_YSEP_PAGE_TOP]
1361   ) then
1362       table.insert(
1363           PROBLEM_REPORT_TABLE,
1364           get_data_page_number(item_data)
1365           .. ' ' .. ITEM_PASSED_YSEP_PAGE_TOP_MESSAGES[item_data[KEY_TYPE]]
1366       )
1367   end
1368
1369   for i=2,#item_data_list do
1370       local item_data = item_data_list[i]
1371       local prev_item_data = item_data_list[i-1]
1372       if (
1373           get_y_item_top(item_data) > get_y_item_bottom(prev_item_data)
1374                               - ysep_list[i-1]
1375       ) then
1376           table.insert(
1377               PROBLEM_REPORT_TABLE,
1378               get_data_page_number(item_data)
1379               .. ' Clash: ' ..
1380               ITEM_CLASH_MESSAGES[prev_item_data[KEY_TYPE]][item_data[KEY_TYPE]]
1381           )
1382       end
1383   end
```

If any item fails to obey ysep page bottom, the last one in the list does.

```
1384   item_data = item_data_list[#item_data_list]
1385   if (
1386       get_y_item_bottom(item_data) < page_data[KEY_VOFFSET]
1387                               + item_data[KEY_YSEP_PAGE_BOTTOM]
1388   ) then
1389       table.insert(
1390           PROBLEM_REPORT_TABLE,
1391           get_data_page_number(item_data)
1392           .. ' ' .. ITEM_PASSED_YSEP_PAGE_BOTTOM_MESSAGE[item_data[KEY_TYPE]]
1393       )
1394   end
1395
1396   end
```

(End of definition for *check_items_vertical*.)

12.12.5 Core vertical position computation

`compute_items_vertical` For every `item_data` in `item_data_list`, compute the field relevant to vertical positioning, namely `KEY_YSHIFT_COMPUTED`, based on the layout information in `page_data`. This may involve setting the field `KEY_ENABLED_COMPUTED` to false. In such a case, the relevant `item_data` is removed from `item_data_list`.

```
1397 local function compute_items_vertical(item_data_list,page_data)
```

Set `KEY_YSHIFT_COMPUTED` of each `item_data` to the user-supplied value.

```
1398   for i=1,#item_data_list do
1399     local item_data = item_data_list[i]
1400
1401     item_data[KEY_YSHIFT_COMPUTED] = item_data[KEY_YSHIFT]
1402   end
```

Decide which items of `TYPE_OPTFIXED` are to be disabled.

```
1403   compute_items_vertical_optfixed_enabled(item_data_list)
```

Strip any `item_data` with `KEY_ENABLED_COMPUTED` set to false from `item_data_list`.

```
1404   list_filter(item_data_list,function(item_data)
1405     return item_data[KEY_ENABLED_COMPUTED]
1406   end)
```

Sort `item_data_list` according to the stored position from top to bottom and left to right on the page, resolving ties using `KEY_ITEMNO`.

```
1407   table.sort(
1408     item_data_list,
1409     function(left,right)
1410       local y_diff = left[KEY_YPOS] - right[KEY_YPOS]
1411
1412       if y_diff > 0 then
1413         return true
1414       elseif y_diff < 0 then
1415         return false
1416       end
1417
1418       local x_diff = left[KEY_XPOS] - right[KEY_XPOS]
1419
1420       if x_diff < 0 then
1421         return true
1422       elseif x_diff > 0 then
1423         return false
1424       end
1425
1426       return (left[KEY_ITEMNO] < right[KEY_ITEMNO])
1427     end
1428   )
```

```
1429   compute_items_vertical_adjustment(item_data_list,page_data)
```

```
1432   check_items_vertical(item_data_list,page_data)
```

```
1433
1434 end
```

(End of definition for `compute_items_vertical`.)

`compute_items` For every item represented in `ITEM_DATA_MAIN_TABLE`, use the `page_data` stored in `PAGE_DATA_MAIN_TABLE` to compute the `item_data` values necessary to place the item correctly on the page, namely those indexed by: `KEY_COLNO_COMPUTED`, `KEY_XSHIFT_COMPUTED`, `KEY_YSHIFT_COMPUTED`, `KEY_SIDE_COMPUTED`, `KEY_ENABLED_COMPUTED`.

```
1435 local function compute_items()
```

Compute the maximum `abspageno`, which will be the last page of the document on which a item appears.

```
1436   local max_abspageno = 0
```

```
1437
```

```
1438   for k,v in pairs(ITEM_DATA_MAIN_TABLE) do
```

```
1439     max_abspageno = math.max(v[KEY_ABSPAGENO],max_abspageno)
```

```
1440   end
```

`per_abspage_item_data_list` will be a list indexed by absolute page numbers. Each entry will be a list (possibly empty) of `item_data` describing the items that appear on the corresponding page.

```
1441   local per_abspage_item_data_list = {}
```

Prepare `per_abspage_item_data_list` by making each entry an empty list, then fill it from `ITEM_DATA_MAIN_TABLE`.

```
1442   for i=1,max_abspageno do
```

```
1443     per_abspage_item_data_list[i] = {}
```

```
1444   end
```

```
1445   for _,item_data in pairs(ITEM_DATA_MAIN_TABLE) do
```

```
1446     local temp_table = per_abspage_item_data_list[item_data[KEY_ABSPAGENO]]
```

```
1447     temp_table[#temp_table+1] = item_data
```

```
1448   end
```

`per_abspage_item_data_list` will be a list indexed by absolute page numbers. Each entry will be a `page_data` describing the corresponding page. Usually multiple entries will be the same `page_data`: in the loop, `pagedatano` will be the index of the last entry in `PAGE_DATA_MAIN_TABLE` with `KEY_ABSPAGENO` value less than or equal to `abspageno`. (There may be several such entries in `PAGE_DATA_MAIN_TABLE` because `\marginalianewgeometry` may have been called multiple times on the same page.) Note that `PAGE_DATA_MAIN_TABLE[0]` is available even if there was no data in the `.aux` file, because the defaults were stored by `store_default_page_data`.

```
1449   local per_abspage_page_data_list = {}
```

```
1450   local pagedatano = 0
```

```
1451   for abspageno = 1,max_abspageno do
```

```
1452     while (
```

```
1453       PAGE_DATA_MAIN_TABLE[pagedatano+1] ~= nil
```

```
1454       and
```

```
1455       PAGE_DATA_MAIN_TABLE[pagedatano+1][KEY_ABSPAGENO] == abspageno
```

```
1456     ) do
```

```
1457       pagedatano = pagedatano+1
```

```
1458     end
```

```
1459     per_abspage_page_data_list[abspageno] = PAGE_DATA_MAIN_TABLE[pagedatano]
```

```
1460   end
```

Iterate through all pages and perform the necessary computations.

```

1461 for abspageno=1,#per_abspage_item_data_list do
1462   local current_page_data = per_abspage_page_data_list[abspageno]
1463   local current_page_item_data_list = per_abspage_item_data_list[abspageno]

```

First, compute the horizontal positions, which includes sorting items into columns in two-column mode.

```

1464   compute_items_horizontal(current_page_item_data_list,current_page_data)

```

Sort the items into sublists corresponding to the margins in which they are located.

```

1465   local current_page_item_data_sublists = {}
1466
1467   for i=0,5 do
1468     current_page_item_data_sublists[i] = {}
1469   end
1470
1471   for _,item_data in pairs(current_page_item_data_list) do
1472     table.insert(
1473       current_page_item_data_sublists[item_data[KEY_MARGINNO_COMPUTED]],
1474       item_data
1475     )
1476   end

```

Compute vertical positons for each sublist.

```

1477   for i=0,5 do
1478     compute_items_vertical(
1479       current_page_item_data_sublists[i],
1480       current_page_data
1481     )
1482   end
1483 end
1484 end

```

(End of definition for compute_items.)

12.13 Passing item_data back to L^AT_EX

`load_item_data` Set the relevant L^AT_EX counter and dimension variables to the values computed for itemno.

```

1485 local function load_item_data(itemno)
1486
1487   item = ITEM_DATA_MAIN_TABLE[tonumber(itemno)]
1488   if item == nil then
1489     item = ITEM_DATA_DEFAULTS
1490   end
1491
1492   tex.count['l__marginalia_page_int'] = item[KEY_PAGENO]
1493   tex.count['l__marginalia_column_computed_int'] = item[KEY_COLNO_COMPUTED]
1494   tex.dimen['l__marginalia_xshift_computed_dim'] = item[KEY_XSHIFT_COMPUTED]
1495   tex.dimen['l__marginalia_yshift_computed_dim'] = item[KEY_YSHIFT_COMPUTED]
1496   tex.count['l__marginalia_side_computed_int'] = item[KEY_SIDE_COMPUTED]
1497   tex.count['l__marginalia_marginno_computed_int']
1498     = item[KEY_MARGINNO_COMPUTED]
1499   if item[KEY_ENABLED_COMPUTED] then
1500     tex.count['l__marginalia_enabled_computed_int'] = 1

```

```

1501 else
1502     tex.count['l__marginalia_enabled_computed_int'] = 0
1503 end
1504
1505 end

```

(End of definition for load_item_data.)

12.14 Export public functions

Finally, make available the functions that will be called from L^AT_EX using `\lua_now:n` and `\lua_now:e`.

```

1506 marginalia = marginalia or {}
1507 marginalia.store_default_page_data = store_default_page_data
1508 marginalia.store_page_data = store_page_data
1509 marginalia.check_page_data = check_page_data
1510
1511 marginalia.store_item_data = store_item_data
1512 marginalia.check_item_data = check_item_data
1513
1514 marginalia.compute_items = compute_items
1515
1516 marginalia.load_item_data = load_item_data
1517
1518 marginalia.write_problem_report = write_problem_report
1519
1520 marginalia.write_page_change_report = write_page_change_report
1521 marginalia.write_item_change_report = write_item_change_report
1522  $\langle$ /lua $\rangle$ 

```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols	
\'	365, 390, 417, 442, 451, 456, 463, 464, 465, 466, 479, 480, 549, 551,
\\	553, 555, 560, 562, 564, 566, 571, 573, 575, 577, 582, 584, 586, 588, 593, 595, 597, 599, 604, 606, 608, 610
B	
\baselineskip	37, 657
\begin	7, 9, 11, 13
bool commands:	
\bool_new:N	652
\bool_set:Nn	655
\bool_to_str:n	410
box commands:	
\box_dp:N	492, 618, 628, 630
\box_ht:N	490, 618, 629
\box_move_up:nn	636
\box_new:N	423
\box_set_dp:Nn	620, 630
\box_set_ht:Nn	619, 629
\box_use:N	638
\l_tmpa_box	626, 628
C	
check commands:	
check_data	795
check_item_data	952
check_items_vertical	1347
check_page_data	893
check_removed_data	828
column (option)	8
\columnsep	409
\columnwidth	408
compute commands:	
compute_items	1435
compute_items_horizontal	1088
compute_items_vertical	1397
compute_items_vertical_adjustment	1269
compute_items_vertical_optfixed_enabled	1222
cs commands:	
\cs_new:Nn	31
\cs_new:Npn	14, 26, 38, 45, 141, 145, 300, 304, 308, 312, 316, 320, 324, 328, 332, 336, 340, 342, 349, 369, 381, 391, 433, 543, 615, 622, 632, 642, 646, 653
\cs_set_eq:NN	49, 52, 346, 353, 359, 362,
D	
\currentgrouplevel	40
D	
dim commands:	
\dim_new:N	78, 79, 80, 81, 82, 83, 150, 151, 152, 153, 200, 201, 202, 203, 204, 205, 424, 425, 428, 429
\dim_set:Nn	35, 489, 491, 618, 628
\dim_set_eq:NN	476, 477
\l_tmpa_dim	618, 619, 620, 628, 629, 630
E	
\end	14
\evensidemargin	50, 405
exp commands:	
\exp_not:N	498, 499, 501, 502
G	
get commands:	
get_data_page_number	762
get_y_item_bottom	1201
get_y_item_top	1196
get_ysep_list	1206
group commands:	
\group_begin:	436, 482
\group_end:	486, 541
H	
\hbox	34
hbox commands:	
\hbox_overlap_left:n	529, 534
\hbox_overlap_right:n	518, 522
\headheight	11, 143, 147
\headsep	11, 143, 147
\hoffset	50, 401
hook commands:	
\hook_gput_code:nnn	33, 50, 356, 361, 378, 414
\hsize	12, 33, 476, 477
I	
\ignorespaces	483
info	685

int commands:

- \int_case:nn 461, 545
- \int_eval:n 400, 498
- \int_gincr:N 393, 435
- \int_if_zero:nTF ... 40, 371, 514, 516
- \int_new:N 54, 62, 70, 128, 389, 422, 426, 427, 430, 431, 432
- \int_set:Nn 58, 66, 74, 395, 396
- \int_set_eq:NN 132
- \int_value:w 399, 401, 402, 403, 404, 405, 406, 407, 408, 409, 439, 497, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511
- \l_tmpa_int 395, 396, 407

iow commands:

- \iow_now:Nn 383, 397
- \iow_shipout_e:Nn 495

K

\kern 519, 524, 531, 536

keys commands:

- \l_keys_choice_int ... 58, 66, 74, 132
- \keys_define:nn 55, 63, 71, 84, 129, 136, 154, 206, 250, 270, 290
- \keys_set:nn 42, 47, 99, 109, 115, 121, 165, 171, 179, 185, 191, 221, 231, 237, 243, 259, 279, 437

L

\lastxpos 501

\lastypos 502

\leavevmode 448

legacy commands:

- \legacy_if:nTF 394, 447
- \legacy_if_gset:nn 658
- \legacy_if_p:n 410, 655

\linewidth 33, 477

list commands:

- list_filter 742, 759

\llap 4, 14

load commands:

- load_item_data 1485

lua commands:

- \lua_now:n .. 27, 61, 299, 302, 306, 310, 314, 318, 322, 326, 330, 334, 338

M

\marginalia 4–9, 13, 14, 16–18, 20, 26, 29, 31, 40, 41, 660

marginalia internal commands:

- \l_marginalia_aux_iow 359, 383, 397, 495
- \l_marginalia_column_computed_int 32, 427, 480
- \l_marginalia_column_int .. 70, 506
- \l_marginalia_default_yshift_dim 136, 507
- \l_marginalia_enabled_computed_int 32, 432, 514
- \l_marginalia_item_box 33, 37, 423, 470, 490, 492, 638
- _marginalia_item_box_set:Nn .. 463, 464, 465, 466, 470
- \l_marginalia_item_depth_dim .. 424, 491, 504
- \l_marginalia_item_height_dim . 424, 489, 503
- \g_marginalia_itemno_int 29, 371, 422, 435, 439, 497
- _marginalia_lua_check_item_data:n 29, 300, 320, 367
- _marginalia_lua_check_page_data:n 29, 300, 308, 364
- _marginalia_lua_compute_items: 300, 324, 358
- _marginalia_lua_load_item_data:n 31, 336, 336, 438
- _marginalia_lua_store_default_page_data: 300, 300, 357
- _marginalia_lua_store_item_data:n 28, 300, 316, 355
- _marginalia_lua_store_page_data:n 28, 300, 304, 348
- _marginalia_lua_write_item_change_report: 300, 332, 374
- _marginalia_lua_write_page_change_report: 312, 375
- _marginalia_lua_write_problem_report: 300, 328, 373
- _marginalia_margin_bottom: ... 141, 145, 187, 198
- _marginalia_margin_top: 141, 141, 181, 197
- \l_marginalia_marginno_computed_int 35, 431, 545
- \l_marginalia_mark_left_between_tl 270, 611
- \l_marginalia_mark_recto_inner_tl 270, 567
- \l_marginalia_mark_recto_outer_tl 270, 556
- \l_marginalia_mark_right_between_tl 270, 600
- \l_marginalia_mark_tl . 520, 522, 532, 534, 555, 566, 577, 588, 599, 610
- \l_marginalia_mark_verso_inner_tl 270, 589

<code>\l_marginalia_mark_verso_outer_-</code> <code>tl</code>	270 , 578	<code>_marginalia_typeset_hmode:n</code> ..	453 , 646
<code>\l_marginalia_max_item_data_-</code> <code>change_int</code>	290	<code>_marginalia_typeset_mmode:n</code> ..	444 , 642 , 642
<code>\l_marginalia_max_page_data_-</code> <code>change_int</code>	290	<code>_marginalia_typeset_vmode:n</code> ..	458 , 642 , 653
<code>\l_marginalia_max_problem_int</code> ..	290	<code>\l_marginalia_valign_int</code> 33 , 128 , 461	
<code>\l_marginalia_nobreak_bool</code>	652 , 655 , 658	<code>_marginalia_vbox_set_center:Nn</code> 615	
<code>\l_marginalia_page_int</code> .. 32 , 426 , 479		<code>_marginalia_vbox_set_midway:Nn</code> 622	
<code>\g_marginalia_pagedatano_int</code> ..	389 , 393 , 399	<code>\l_marginalia_width_dim</code> 33 , 35 , 476 , 551 , 562 , 573 , 584 , 595 , 606	
<code>_marginalia_place_item_box:</code> ..	34 , 525 , 530 , 632 , 632	<code>\l_marginalia_width_left_-</code> <code>between_dim</code>	200 , 607
<code>\l_marginalia_pos_int</code> 62 , 505		<code>\l_marginalia_width_recto_-</code> <code>inner_dim</code>	200 , 563
<code>_marginalia_process_item:nn</code> ..	34 , 38 , 433 , 433 , 662	<code>\l_marginalia_width_recto_-</code> <code>outer_dim</code>	200 , 552
<code>_marginalia_process_item_-</code> <code>data:n</code>	28 , 29 , 351 , 354 , 366	<code>\l_marginalia_width_right_-</code> <code>between_dim</code>	200 , 596
<code>_marginalia_process_page_-</code> <code>data:n</code>	28 , 29 , 344 , 347 , 363	<code>\l_marginalia_width_verso_-</code> <code>inner_dim</code>	200 , 585
<code>_marginalia_set_dim:Nn</code> .. 20 , 31 , 31 , 87 , 89 , 91 , 93 , 95 , 97 , 157 , 159 , 161 , 163 , 209 , 211 , 213 , 215 , 217 , 219		<code>\l_marginalia_width_verso_-</code> <code>outer_dim</code>	200 , 574
<code>_marginalia_set_xsep_width_-</code> <code>style_mark:</code>	468 , 543 , 543	<code>_marginalia_write_page_data:</code> ..	30 , 390 , 390 , 418 , 420 , 670
<code>_marginalia_setup:n</code>	20 , 38 , 49 , 49 , 52 , 666	<code>_marginalia_write_page_data_-</code> <code>real:</code>	391 , 419
<code>_marginalia_setup_body:n</code>	20 , 45 , 45 , 52	<code>_marginalia_write_reports:</code>	369 , 369 , 379
<code>_marginalia_setup_preamble:n</code> ..	20 , 38 , 38 , 49	<code>_marginalia_write_version:</code>	381 , 381 , 416
<code>\l_marginalia_side_computed_int</code>	32 , 34 , 430 , 516	<code>\l_marginalia_xsep_dim</code> .. 34 , 35 , 524 , 531 , 549 , 560 , 571 , 582 , 593 , 604	
<code>\l_marginalia_style_left_-</code> <code>between_tl</code>	250 , 609	<code>\l_marginalia_xsep_left_-</code> <code>between_dim</code>	78 , 605
<code>\l_marginalia_style_recto_-</code> <code>inner_tl</code>	250 , 565	<code>\l_marginalia_xsep_recto_inner_-</code> <code>dim</code>	78 , 561
<code>\l_marginalia_style_recto_-</code> <code>outer_tl</code>	250 , 554	<code>\l_marginalia_xsep_recto_outer_-</code> <code>dim</code>	78 , 550
<code>\l_marginalia_style_right_-</code> <code>between_tl</code>	250 , 598	<code>\l_marginalia_xsep_right_-</code> <code>between_dim</code>	78 , 594
<code>\l_marginalia_style_tl</code>	35 , 475 , 553 , 564 , 575 , 586 , 597 , 608	<code>\l_marginalia_xsep_verso_inner_-</code> <code>dim</code>	78 , 583
<code>\l_marginalia_style_verso_-</code> <code>inner_tl</code>	250 , 587	<code>\l_marginalia_xsep_verso_outer_-</code> <code>dim</code>	78 , 572
<code>\l_marginalia_style_verso_-</code> <code>outer_tl</code>	250 , 576	<code>\l_marginalia_xshift_computed_-</code> <code>dim</code>	32 , 34 , 428 , 519 , 536
<code>_marginalia_tagging_socket:n</code> ..	19 , 14 , 26 , 469 , 472 , 488	<code>\l_marginalia_ysep_above_dim</code> ..	150 , 508
<code>\l_marginalia_type_int</code> 54 , 500		<code>\l_marginalia_ysep_below_dim</code> ..	150 , 509
<code>_marginalia_typeset:n</code>	443 , 452 , 457 , 493	<code>\l_marginalia_ysep_page_bottom_-</code> <code>dim</code>	150 , 511
<code>_marginalia_typeset_hmmode:n</code> ..	642		

shipout commands:		<code>\UseTaggingSocket</code> 19, 28
<code>\g_shipout_readonly_int</code> ... 400, 498		
skip commands:		V
<code>\skip_vertical:n</code> 657		<code>valign</code> (option) 9
<code>\smash</code> 37, 634		<code>\vbox</code> 36
socket commands:		vbox commands:
<code>\socket_new:nn</code> 19, 20, 24		<code>\vbox_set:Nn</code> 464, 626
store commands:		<code>\vbox_set_center:Nn</code> 465, 615
<code>store_default_page_data</code> 881		<code>\vbox_set_midway:Nn</code> 466, 622
<code>store_item_data</code> 943		<code>\vbox_set_top:Nn</code> 463, 617, 624
<code>store_page_data</code> 872		<code>\voffset</code> 11, 143, 147, 402
str commands:		<code>\vtop</code> 36
<code>\str_if_exist:NTF</code> 17, 22		
<code>\str_use:N</code> 500		W
style (option) 13		warning 685
style left between (option) 13		width (option) 12
style recto inner (option) 13		width between (option) 12
style recto outer (option) 13		width inner (option) 12
style right between (option) 13		width left between (option) 12
style verso inner (option) 13		width outer (option) 12
style verso outer (option) 13		width recto inner (option) 12
sys commands:		width recto outer (option) 12
<code>\sys_if_engine luatex:TF</code> 6		width right between (option) 12
	T	width verso inner (option) 12
tag commands:		width verso outer (option) 12
<code>\tag_resume:n</code> 627		write commands:
<code>\tag_suspend:n</code> 625		<code>write_item_change_report</code> 1004
TeX and L ^A T _E X 2 _ε commands:		<code>write_page_change_report</code> 1016
<code>\@bsphack</code> 37, 648		<code>write_problem_report</code> 992
<code>\@esphack</code> 37, 650		<code>write_report</code> 960
<code>\@ifundefined</code> 12		
<code>\@mainaux</code> 359		X
<code>\@parboxrestore</code> 33, 471		<code>xsep</code> (option) 9
<code>\c@page</code> 499		<code>xsep between</code> (option) 9
<code>\if@nobreak</code> 37		<code>xsep inner</code> (option) 9
<code>\marginalia@itemdata</code> . 28, 29, 349, 496		<code>xsep left between</code> (option) 9
<code>\marginalia@notedata</code> 42		<code>xsep outer</code> (option) 9
<code>\marginalia@pagedata</code> 28, 29, 42, 340, 342, 398		<code>xsep recto inner</code> (option) 9
<code>\marginalia@version</code> 340, 384		<code>xsep recto outer</code> (option) 9
<code>\textheight</code> 11, 148		<code>xsep right between</code> (option) 9
<code>\textwidth</code> 406		<code>xsep verso inner</code> (option) 9
tl commands:		<code>xsep verso outer</code> (option) 9
<code>\tl_if_empty:NTF</code> 520, 532		
<code>\tl_use:N</code> 475, 522, 534		Y
token commands:		<code>ysep</code> (option) 11
<code>\token_to_str:N</code> 384, 398, 496		<code>ysep above</code> (option) 11
<code>\topmargin</code> 11, 143, 147		<code>ysep below</code> (option) 11
<code>\twocolumn</code> 5		<code>ysep page bottom</code> (option) 11
type (option) 7		<code>ysep page bottom margin</code> (option) 11
	U	<code>ysep page top</code> (option) 11
use commands:		<code>ysep page top bottom margin</code> (option) . 11
<code>\use:N</code> 385		<code>ysep page top margin</code> (option) 11
		<code>yshift</code> (option) 11