

Parallel typesetting for critical editions: the **eledpar** package*

Peter Wilson
Herries Press[†]
Maïeul Rouquette[‡]

Abstract

The **eledmac** package, which is based on the PLAIN T_EX set of EDMAC macros, has been used for some time for typesetting critical editions. The **eledpar** package is an extension to **eledmac** which enables texts and their critical apparatus to be typeset in parallel, either in two columns or on pairs of facing pages.

Note that before September 2012, **eledpar** was called **ledpar**. The changes from **ledmac**/**ledpar** to **eledmac**/**eledpar** is explained in **ledmac** documentation.

eledpar provides many tools and options. Normally, they are all documented in this file. Also provided is a help folder, “examples”. The folder contains additional examples (although not for all cases).

To report bugs, please go to **ledmac**’s GitHub page and click “New Issue”: <https://github.com/maieul/ledmac/issues/>. You must open an account with github.com to access my page (maieul/ledmac). GitHub accounts are free for open-source users. You can report bug in English or in French (better).

You can subscribe to the **eledmac** email list in:
<http://geekographie.maieul.net/146>

Contents

1 Introduction	4
2 The eledpar package	4
2.1 General	5
3 Parallel columns	5

*This file (**eledpar.dtx**) has version number v1.13.1, last revised 2015/03/12.

[†]herries dot press at earthlink dot net

[‡]maieul at maieul dot net

4 Facing pages	7
4.1 Basic usage	7
4.2 Text width	7
4.3 Setting	7
4.4 Critical and familiar footnotes	8
4.4.1 Note size setting	8
4.4.2 Notes for one side only	8
4.4.3 Familiar notes called in the right side, but to be printed in the left side	8
5 Left and right texts	9
6 Numbering text lines and paragraphs	10
7 Verse	12
8 Side notes	14
9 Parallel ledgroups	14
9.1 Parallel ledgroups and <code>setspace</code> package	15
10 Sectioning commands	15
11 Implementation overview	16
12 Preliminaries	16
12.1 Messages	17
13 Sectioning commands	18
14 Line counting	20
14.1 Choosing the system of lineation	20
14.2 Line-number counters and lists	24
14.3 Reading the line-list file	24
14.4 Commands within the line-list file	26
14.5 Writing to the line-list file	33
15 Marking text for notes	36
15.1 Specific hooks and commands for notes	36
15.1.1 Create commands to memorize display options	37
15.1.2 Boolean flags for notes to be printed in one side only	37
15.1.3 Familiar footnotes without marks	37
15.1.4 Create hooks	38
15.1.5 Init standards series (A,B,C,D,E,Z)	38
16 Pstart numbers dumping and restoration	38
17 Parallel environments	39

18 Paragraph decomposition and reassembly	41
18.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	42
18.2 Processing one line	46
18.3 Line and page number computation	49
18.4 Line number printing	52
18.5 Pstart number printing in side	54
18.6 Add insertions to the vertical list	55
18.7 Penalties	56
18.8 Printing leftover notes	57
19 Footnotes	57
19.1 Normal footnote formatting	57
19.2 Footnotes output specific to <code>\Pages</code>	58
20 Cross referencing	62
21 Side notes	62
22 Familiar footnotes	64
23 Verse	65
24 Naming macros	67
25 Counts and boxes for parallel texts	67
26 Fixing babel	69
27 Parallel columns	72
28 Parallel pages	77
29 Sections' titles' commands	89
30 Page break/no page break, depending on the specific line	90
31 Parallel ledgroup	90
32 The End	93
Appendix A Some things to do when changing version	94
Appendix A.1 Migration to <code>eledpar</code> 1.4.3	94
References	94
Index	94
Change History	105

1 Introduction

The EDMAC macros [LW90] for typesetting critical editions of texts have been available for use with TeX for some years. Since EDMAC became available there had been a small but constant demand for a version of EDMAC that could be used with L^AT_EX. The *eledmac* package was introduced in 2003 in an attempt to satisfy that request.

Some critical editions contain texts in more than one form, such as a set of verses in one language and their translations in another. In such cases there is a desire to be able to typeset the two texts, together with any critical apparatus, in parallel. The *eledpar* package is an extension to *eledmac* that enables two texts and their apparatus to be set in parallel, either in two columns or on pairs of facing pages.

The package has to try and coerce T_EX into paths it was not designed for. Use of the package, therefore, may produce some surprising results.

This manual contains a general description of how to use *eledpar* starting in section 2; the complete source code for the package, with extensive documentation (in sections 11 through 32); and an Index to the source code. As *eledpar* is an adjunct to *eledmac* I assume that you have read the *eledmac* manual. Also *eledpar* requires *eledmac* to be used, preferably at least version 0.10 (2011/08/22). You do not need to read the source code for this package in order to use it but doing so may help to answer any questions you might have. On a first reading, I suggest that you should skip anything after the general documentation in sections 2 until 11, unless you are particularly interested in the innards of *eledpar*.

2 The *eledpar* package

A file may mix *numbered* and *unnumbered* text. Numbered text is printed with marginal line numbers and can include footnotes and endnotes that are referenced to those line numbers: this is how you'll want to print the text that you're editing. Unnumbered text is not printed with line numbers, and you can't use *eledmac*'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

The *eledpar* package lets you typeset two *numbered* texts in parallel. This can be done either as setting the 'Leftside' and 'Rightside' texts in two columns or on facing pages. In the paired pages case footnotes are placed at the bottom of the page on which they are called out — that is, footnotes belonging to the left are set at the foot of a left (even numbered) page, and those for right texts are at the bottom of the relevant right (odd numbered) page. However, in the columnar case, all footnotes are set at the bottom left of the page on which they are called out — they are not set below the relevant column. The line numbering schemes need not be the same for the two texts.

2.1 General

`eledmac` essentially puts each chunk of numbered text (the text within a `\pstart` ... `\pend`) into a box and then following the `\pend` extracts the text line by line from the box to number and print it. More precisely, the text is first put into the box as though it was being typeset as normal onto a page and any notes are stored without being typeset. Then each typeset line is extracted from the box and any notes for that line are recalled. The line, with any notes, is then output for printing, possibly with a line number attached. Effectively, all the text is typeset and then afterwards all the notes are typeset.

`eledpar` similarly puts the left and right chunks into boxes but can't immediately output the text after a `\pend` — it has to wait until after both the left and right texts have been collected before it can start processing. This means that several boxes are required and possibly TeX has to store a lot of text in its memory; both the number of potential boxes and memory are limited. If TeX's memory is overfilled the recourse is to reduce the amount of text stored before printing.

`\maxchunks` It is possible to have multiple chunks in the left and right texts before printing them. The macro `\maxchunks{<num>}` specifies the maximum number of chunks within the left or right texts. This is initially set as:

```
\maxchunks{5120}
```

meaning that there can be up to 5120 chunks in the left text and up to 5120 chunks in the right text, requiring a total of 10240 boxes. If you need more chunks then you can increase `\maxchunks`. The `\maxchunks` must be called in the preamble.

TeX has a limited number of boxes; if you get an error message along the lines of 'no room for a new box', then load the package `etex`, which needs `pdflatex` or `xelatex`. If you `\maxchunks` is too little you can get a `eledmac` error message along the lines: 'Too many `\pstart` without printing. Some text will be lost.' then you will have to either increase `\maxchunks` or use the parallel printing commands (`\Columns` or `\Pages`) more frequently.

When typesetting verse using `\syntax`, each line is treated as a chunk, so be warned that if you are setting parallel verses you might have to increase `\maxchunks` much more than it appears at first sight.

In general, `eledmac` is a TeX resource hog, and `eledpar` only makes things worse in this respect.

3 Parallel columns

`pairs` Numbered text that is to be set in columns must be within a `pairs` environment. Within the environment the text for the lefthand and righthand columns is placed within the `Leftside` and `Rightside` environments, respectively; these are described in more detail below in section 5.

`\Columns` The command `\Columns` typesets the texts in the previous pair of `Leftside` and `Rightside` environments. The general scheme for parallel columns looks like this:

```
\begin{pairs}
```

```

\begin{Leftside} ... \end{Leftside}
\begin{Rightside} ... \end{Rightside}
\Columns
\begin{Leftside} ... \end{Leftside}
...
\end{pairs}
\Columns

```

Keep in mind that the `\Columns` **must be** outside of the `pairs` environment. You can use the macro `\AtBeginPairs` to insert a code at the beginning of each `pairs` environments. That could be useful to add the `\sloppy` macro to prevent overfull hboxes in two columns.

```
\AtBeginPairs{\sloppy}
```

There is no required pagebreak before or after the columns.

`\Lcolwidth` The lengths `\Lcolwidth` and `\Rcolwidth` are the widths of the left and right columns, respectively. By default, these are:

```
\setlength{\Lcolwidth}{0.45\textwidth}
```

```
\setlength{\Rcolwidth}{0.45\textwidth}
```

They may be adjusted if one text tends to be ‘bulkier’ than the other.

`\columnrulewidth` The macro `\columnseparator` is called between each left/right pair of lines. By default it inserts a vertical rule of width `\columnrulewidth`. As this is initially defined to be 0pt the rule is invisible. For a visible rule between the columns you could try:

```
\setlength{\columnrulewidth}{0.4pt}
```

You can also modify `\columnseparator` if you want more control.

`\columnspan` By default, columns are positioned to the right of the page. However, you use `\columnspan{L}` to align them to the left, or `\columnspan{C}` to center them.

When you use `\stanza`, the visible rule may shift when a verse has a hanging indent. To prevent shifting, use `\setstanzaindent` outside the `Leftside` or `Rightside` environment.

`\beforecolumnseparator` By default, the spaces around column separator are the same as the space:

`\aftercolumnseparator`

- On the left of columns, if columns are aligned right.
- On the right of columns, if columns are aligned left.
- On both the Left and Right columns, if columns are centered.

You can redefine `\beforecolumnseparator` and `\aftercolumnseparator` length to define spaces before or after the column separator, instead of letting `eledpar` calculate them automatically.

```
\setlength{\beforecolumnseparator}{length}
```

```
\setlength{\aftercolumnseparator}{length}
```

`\widthliketwocolumns` If you want to come back to the previous behavior, just set them with a negative value. If you want to mix texts in columns and text without columns, you can horizontally align text in one column to text in two columns with `\widthliketwocolumnstrue`. To reset this feature, just use `\widthliketwocolumnsfalse`.

`\Xnoteswidthliketwocolumns`
`\notesXwidthliketwocolumns` In most cases, you should use `\widthliketwocolumns` in combination with `\Xnoteswidthliketwocolumns` and `\notesXwidthliketwocolumns` to align the critical/familiar footnotes with the two columns. See `eledmac`'s handbook for more details.

4 Facing pages

4.1 Basic usage

`pages` Numbered text that is to be set on facing pages must be within a `pages` environment. Within the environment the text for the lefthand and righthand pages is placed within the `Leftside` and `Rightside` environments, respectively.

`\Pages` The command `\Pages` typesets the texts in the previous pair of `Leftside` and `Rightside` environments. The general scheme for parallel pages looks like this:

```
\begin{pages}
\begin{Leftside} ... \end{Leftside}
\begin{Rightside} ... \end{Rightside}
\begin{Leftside} ... \end{Leftside}
...
\end{pages}
\Pages
```

The `Leftside` text is set on lefthand (even numbered) pages and the `Rightside` text is set on righthand (odd numbered) pages. Each `\Pages` command starts a new even numbered page. After parallel typesetting is finished, a new page is started. Note that the `\Pages` **must be** outside of the `pages` environment.

4.2 Text width

`\Lcolwidth`
`\Rcolwidth` Within the `pages` environment the lengths `\Lcolwidth` and `\Rcolwidth` are the widths of the left and right pages, respectively. By default, these are set to the normal `textwidth` for the document, but can be changed within the environment if necessary.

4.3 Setting

`\goalfraction` When doing parallel pages `eledpar` has to guess where TeX is going to put page-breaks and hopefully get there first in order to put the pair of texts on their proper pages. When it thinks that the fraction `\goalfraction` of a page has been filled, it finishes that page and starts on the other side's text. The definition is:

```
\newcommand*{\goalfraction}{0.9}
```

If you think you can get more on a page, increase this. On the other hand, if some left text overflows onto an odd numbered page or some right text onto an even page, try reducing it, for instance by:

```
\renewcommand*{\goalfraction}{0.8}
```

4.4 Critical and familiar footnotes

Of course, in “Facing pages”, the `eledmac` both critical and familiar footnotes can be used. However, some specific points must be taken into consideration.

4.4.1 Note size setting

Since `eledpar` v.1.13.0, long notes in facing pages can flow from left to right pages, and *vice-versa*. However, the `eledmac` default setting for the maximum allotted size to notes is greater than `\textheight`. That makes impossible for long notes to flow across pages.¹ We have not changed this default setting, because we don’t want to break compatibility with older version of `eledmac`. So, you MUST change the default setting via `\maxhXnotes` (for critical notes) `\maxhnotesX` (for familiar notes). Both commands are explained in handbook (4.4.9 p. 24). As and advisable setting:

```
\maxhXnotes{0.6\textheight}
\maxhnotesX{0.6\textheight}
```

4.4.2 Notes for one side only

<code>\onlyXside</code> <code>\onlysideX</code>	<p>You may want to typeset notes on one side only (either left or right). Use <code>\onlyXside[⟨s⟩]{⟨p⟩}</code> to set critical notes, and <code>\onlysideX[⟨s⟩]{⟨p⟩}</code> to set familiar notes. <code>{⟨p⟩}</code> must be set to L for notes to be confined only on the left side and to R for notes to be confined only on the right side.</p>
--	--

4.4.3 Familiar notes called in the right side, but to be printed in the left side

<code>\footnoteXnomk</code> <code>\footnoteXmk</code>	<p>As often happens, the left side has less room for text. We may want to call familiar notes in the right side while using at the same time the available space in the left side to print them.</p>
--	--

To achieve this, we call `\footnoteXnomk{⟨notecontent⟩}` in the left side. X is to be replaced by the series letter. We do this call in the left side after the word which matches up to the one in the right side after which we want to insert the actual footnote mark.

In the right side, we call `\footnoteXmk` at the place we want to have the footnote mark. X is to be replaced by the series letter. For example:

¹The same applies to L^AT_EX normal notes. Read <http://tex.stackexchange.com/a/228283/7712> for technical informations.


```

\begin{Leftside}
\beginnumbering
\pstart
  A little cat\footnoteAnomk{A note.}. And so one ...
\pend
\endnumbering
\end{Leftside}
\begin{Rightside}
\beginnumbering
\pstart
  Un petit chat\footnotemk. And so one ...
\pend
\endnumbering
\end{Rightside}

```

5 Left and right texts

Parallel texts are divided into Leftside and Rightside. The form of the contents of these two are independent of whether they will be set in columns or pages.

Leftside The left text is put within the **Leftside** environment and the right text like-
Rightside wise in the **Rightside** environment. The number of **Leftside** and **Rightside** environments must be the same.

Within these environments you can designate the line numbering scheme(s) to be used. The **eledmac** package originally used counters for specifying the numbering scheme; now both **eledmac** and the **eledpar** package use macros instead. Following `\firstlinenum{<num>}` the first line number will be *<num>*, and following `\linenumincrement{<num>}` only every *<num>*th line will have a printed number. Using these macros inside the **Leftside** and **Rightside** environments gives you independent control over the left and right numbering schemes. The `\firstsublinenum` and `\sublinenumincrement` macros correspondingly set the numbering scheme for sublines. The starred versions change both left and right numbering schemes.

`\pstart` In a serial (non-parallel) mode, each numbered paragraph, or chunk, is contained between the `\pstart` and `\pend` macros, and the paragraph is output when the `\pend` macro occurs. The situation is somewhat different with parallel typesetting as the left text (contained within `\pstart` and `\pend` groups within the **Leftside** environment) has to be set in parallel with the right text (contained within its own `\pstart` and `\pend` groups within the corresponding **Rightside** environment) the `\pend` macros cannot immediately initiate any typesetting — this has to be controlled by the `\Columns` or `\Pages` macros. Several chunks may be specified within a **Leftside** or **Rightside** environment. A multi-chunk text then looks like:

```

\begin{...side}
% \beginnumbering
\pstart first chunk \pend

```

```

\pstart  second chunk \pend
...
\pstart  last chunk \pend
% \endnumbering
\end{...side}

```

Numbering, via `\beginnumbering` and `\endnumbering`, may extend across several `Leftside` or `Rightside` environments. Remember, though, that the left/right sides are effectively independent of each other.

Generally speaking, controls like `\firstlinenum` or `\linenummargin` apply to sequential and left texts. To effect right texts only they have to be within a `Rightside` environment. `\lineationR` macro is the equivalent of `eledmac \lineation` macro for the right side. `\lineation*` macro is the equivalent of `eledmac \lineation` macro for both sides. If you are using the `babel` package with different languages (via, say, `\selectlanguage`) for the left and right texts it is particularly important to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default. Also, it is the *last* `\selectlanguage` in a side that controls the language used in any notes for that side when they get printed. If you are using multilingual notes then it is probably safest to explicitly specify the language(s) for each note rather than relying on the language selection for the side. The right side language is also applied to the right side line numbers.

Corresponding left and right sides must have the same number of paragraph chunks — if there are four on the left there must be four on the right, even if some are empty. The start of each pair of left and right chunks are aligned horizontally on the page. The ends may come at different positions — if one chunk is shorter than the other then blank lines are output on the shorter side until the end of the longer chunk is reached.

However, sometime if the left pstarts are much greater than right pstarts, or *vice-versa*, you can decide to shift the pstarts on the left and right side. That means the start of pstarts are not aligned horizontally on the page, the shift is offset at the end of each double pages. To enable this function, load `eledpar` with the option `shiftedpstarts`.

6 Numbering text lines and paragraphs

`\beginnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, like:

```

\beginnumbering
<text>
\endnumbering

```

These have to be separately specified within `Leftside` and `Rightside` environments.

The `\beginnumbering` macro resets the line number to zero, reads an auxiliary file called `<jobname>.nn` (where `<jobname>` is the name of the main input file for

this job, and `nn` is 1 for the first numbered section, 2 for the second section, and so on), and then creates a new version of this auxiliary file to collect information during this run. Separate auxiliary files are maintained for right hand texts and these are named `\jobname.nnR`, using the ‘R’ to distinguish them from the left hand and serial (non-parallel) texts.

`\memorydump` The command `\memorydump` effectively performs an `\endumbering` immediately followed by a `\beginnumbering` while not restarting the numbering sequence. This has the effect of clearing TeX’s memory of previous texts and any associated notes, allowing longer apparent streams of parallel texts. The command should be applied to both left and right texts, and after making sure that all previous notes have been output. For example, along the lines of:

```
\begin{Leftside}
  \beginnumbering
  ...
\end{Leftside}
\begin{Rightside}
  \beginnumbering
  ...
\end{Rightside}
\Pages
\begin{Leftside}
  \memorydump
  ...
\end{Leftside}
\begin{Rightside}
  \memorydump
  ...
```

`\Rlineflag` The value of `\Rlineflag` is appended to the line numbers of the right texts. Its default definition is:

```
\newcommand*\Rlineflag{\R}
```

This may be useful for parallel columns but for parallel pages it might be more appropriate to redefine it as:

`\printlinesR` `\renewcommand*\Rlineflag{\}`. The `\printlines` macro is ordinarily used to print the line number references for critical footnotes. For footnotes from right side texts a special version is supplied, called `\printlinesR`, which incorporates `\Rlineflag`. (The macro `\ledsavedprintlines` is a copy of the original `\printlines`, just in case ...). As provided, the package makes no use of `\printlinesR` but you may find it useful. For example, if you only use the B footnote series in righthand texts then you may wish to flag any line numbers in those footnotes with the value of `\Rlineflag`. You could do this by putting the following code in your preamble:

```
\let\oldBfootfmt\Bfootfmt
\renewcommand{\Bfootfmt}[3]{%
  \let\printlines\printlinesR
  \oldBfootfmt{#1}{#2}{#3}}
```

It's possible to insert a number at every `\pstart` command. You must use the `\numberpstarttrue` command to have it. You can stop the numerotation with `\numberpstartfalse`. You can redefine the commands `\thepstartL` and `\thepstartR` to change style. The numbering restarts on each `\beginnumbering`

7 Verse

If you are typesetting verse with `eledmac` you can use the `\stanza` construct, and you can also use this in right or left parallel texts. In this case each verse line is a chunk which has two implications. (1) you can unexpectedly exceed the `\maxchunks` limit or the overall limit on the number of boxes, and (2) left and right verse lines are matched, which may not be desirable if one side requires more print lines for verse lines than the other does.

`astanza` `eledpar` provides an `astanza` environment which you can use instead of `\stanza` (simply replace `\stanza` by `\begin{astanza}` and add `\end{astanza}` after the ending `\&`). Within the `astanza` environment each verse line is treated as a paragraph, so there must be no blank lines in the environment otherwise there will be some extraneous vertical spacing.

If you get an error message along the lines of ‘Missing number, treated as zero `\sza@00`’ it is because you have forgotten to use `\setstanzaindents` to set the stanza indents.

`\skipnumbering` The command `\skipnumbering` when inserted in a line of parallel text causes the numbering of that particular line to be skipped. This can useful if you are putting some kind of marker (even if it is only a blank line) between stanzas. Remember, parallel texts must be numbered and this provides a way to slip in an ‘unnumbered’ line.

The `astanza` environment forms a chunk but you may want to have more than one stanza within the chunk. Here are a couple of ways of doing that with a blank line between each internal stanza, and with each stanza numbered. First some preliminary definitions:

```
\newcommand*{\stanzanum}[2][\stanzaindentbase]{%
  \hspace{-#1\llap{\textbf{#2}}}\hspace{#1\ignorespaces}
\newcommand{\interstanza}{\par\mbox{}\skipnumbering}
```

And now for two stanzas in one. In this first example the line numbering repeats for each stanza.

```
\setstanzaindents{1,0,1,0,1,0,1,0,1,0,1}
\begin{pairs}
\begin{Leftside}
  \firstlinenum{2}
  \linenumincrement{1}
  \beginnumbering
  \begin{astanza}
```

```

\stanzanum{1} First in first stanza &
                Second in first stanza &
                Second in first stanza &
                Third in first stanza &
                Fourth in first stanza &
\interstanza
\setline{2}\stanzanum{2} First in second stanza &
                Second in second stanza &
                Second in second stanza &
                Third in second stanza &
                Fourth in second stanza \&
\end{astanza}
...

```

And here is a slightly different way of doing the same thing, but with the line numbering being continuous.

```

\setstanzaindents{1,0,1,0,1,0,0,1,0,1,0,1}
\begin{pairs}
\begin{Leftside}
\firstlinenum{2}
\linenumincrement{1}
\beginnumbering
\begin{astanza}
\stanzanum{1} First in first stanza &
                Second in first stanza &
                Second in first stanza &
                Third in first stanza &
                Fourth in first stanza &
\strut &
\stanzanum{2}\advanceline{-1} First in second stanza &
                Second in second stanza &
                Second in second stanza &
                Third in second stanza &
                Fourth in second stanza \&
\end{astanza}
...

```

`\hangingsymbol` Like in `eledmac`, you could redefine the command `\hangingsymbol` to insert a character in each hanging line. If you use it, you must run \LaTeX two time. Example for the French typography

```
\renewcommand{\hangingsymbol}{[,]}
```

You can also use it to force hanging verse to be flush right:

```
\renewcommand{\hangingsymbol}{\protect\hfill}
```

When you use `\lednopb` make sure to use it on both sides in the corresponding verses to keep the pages in sync.

8 Side notes

As in `eledmac`, you must use one of the following commands to add side notes: `\ledsidenote`, `\ledleftnote`, `\ledrightnote`, `\ledouterote`, `\ledinnerrote`.

The `\sidenotemargin` defines the margin of the sidenote for either left or right side, depending on the current environment. You can use `\sidenotemargin*` to define it for both sides.

9 Parallel ledgroups

You can also make parallel ledgroups (see the documentation of `eledmac` about ledgroups). To do it you have:

- To load `eledpar` package with the `parledgroup` option, or to add `\parledgrouptrue`.
- To push each ledgroup between `\pstart... \pend` command.

See the following example:

```
\begin{pages}
\begin{Leftside}
\beginnumbering
\pstart
\begin{ledgroup}
ledgroup content
\end{ledgroup}
\pend
\pstart
\begin{ledgroup}
ledgroup content
\end{ledgroup}
\pend
\endnumbering
\end{Leftside}
\begin{Rightside}
\beginnumbering
\pstart
\begin{ledgroup}
ledgroup content
\end{ledgroup}
\pend
\pstart
\begin{ledgroup}
ledgroup content
\end{ledgroup}
\pend
\endnumbering
\end{Rightside}
\Pages
```

```
\end{pages}
```

You can add sectioning a sectioning command, following this scheme:

```
\begin{..side}
  \beginnumbering
  \pstart
  \section{First ledgroup title}
\pend
  \pstart
  \begin{ledgroup}\skipnumbering
    ledgroup content
  \end{ledgroup}
\pend
  \pstart
  \section{Second ledgroup title}
\pend
  \pstart
  \begin{ledgroup}\skipnumbering
    ledgroup content
  \end{ledgroup}
\pend
\endnumbering
\end{..side}
```

9.1 Parallel ledgroups and setspace package

If you use the `setspace` package and want your notes in parallel ledgroups to be single-spaced (not half-spaced or double-spaced), just add to your preamble:

```
\let\parledgroupnotespacing\singlespacing
```

In effect, to have correct spacing, don't change the font size of your notes.

10 Sectioning commands

`\eledsectnotoc`

The standard sectioning commands of `eledmac` are available, and provide parallel sectionings, for both two-column and two-page layout. By default, the section commands of the right side are not added to the table of contents. But you can change it, using `\eledsectnotoc{<arg>}`, where `<arg>` could be `L` (for left side) or `R` (for right side).

`\eledsectmark`

By default, the `LATEX` marks for header are token from left side. You can change it, using `\eledsectmark{<arg>}`, where `<arg>` could be `L` (for left side) or `R` (for right side).

11 Implementation overview

TeX is designed to process a single stream of text, which may include footnotes, tables, and so on. It just keeps converting its input into a stream typeset pages. It was not designed for typesetting two texts in parallel, where it has to alternate from one to the other. Further, TeX essentially processes its input one paragraph at a time — it is very difficult to get at the ‘internals’ of a paragraph such as the individual lines in case you want to number them or put some mark at the start or end of the lines.

`eledmac` solves the problem of line numbering by putting the paragraph in typeset form into a box, and then extracting the lines one by one from the box for TeX to put them onto the page with the appropriate page breaks. Most of the `eledmac` code is concerned with handling this box and its contents.

`eledpar`’s solution to the problem of parallel texts is to put the two texts into separate boxes, and then appropriately extract the pairs of lines from the boxes. This involves duplicating much of the original box code for an extra right text box. The other, smaller, part of the code is concerned with coordinating the line extractions from the boxes.

The package code is presented in roughly in the same order as in `eledmac`.

12 Preliminaries

Announce the name and version of the package, which is targetted for LaTeX2e. The package also requires the `eledmac` package.

```
1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{eledpar}[2015/03/12 v1.13.1 eledmac extension for parallel texts]%
4
```

Few commands use `\xspace` command.

```
5 \RequirePackage{xspace}%
```

With the option ‘`shiftedpstarts`’ a long `pstart` on the left side (or in the right side) doesn’t make a blank on the corresponding `pstart`, but the blank is put on the bottom of the page. Consequently, the `pstarts` on the parallel pages are shifted, but the shift stops at every end of pages. The `\shiftedverses` is kept for backward compatibility.

```
\ifshiftedpstarts
```

```
6 \newif\ifshiftedpstarts
7 \let\shiftedversestrue\shiftedpstartstrue
8 \let\shiftedversesfalse\shiftedpstartsfalse
9 \DeclareOption{shiftedverses}{\shiftedpstartstrue}
10 \DeclareOption{shiftedpstarts}{\shiftedpstartstrue}
11 \DeclareOption{parledgroup}{\parledgrouptrue}
```

`\ifwidthliketwocolumns` The `\widthliketwocolumns` option can be called both in `eledpar` and `eledmac`.


```
12 \DeclareOption{widthliketwocolumns}{\widthliketwocolumnstrue}%
```

```
13 \ProcessOptions%
```

As noted above, much of the code is a duplication of the original `eledmac` code to handle the extra box(es) for the right hand side text, and sometimes for the left hand side as well. In order to distinguish I use ‘R’ or ‘L’ in the names of macros for the right and left code. The specifics of ‘L’ and ‘R’ are normally hidden from the user by letting the `Leftside` and `Rightside` environments set things up appropriately.

```
\ifl@dpairing \ifl@dpairing is set TRUE if we are processing parallel texts and \ifl@dpaging
\ifl@dpaging is also set TRUE if we are doing parallel pages. \ifledRcol is set TRUE if we
\ifledRcol are doing the right hand text. They are defined in eledmac.
```

```
\Lcolwidth The widths of the left and right parallel columns (or pages).
```

```
\Rcolwidth 14 \newdimen\Lcolwidth
15 \Lcolwidth=0.45\textwidth
16 \newdimen\Rcolwidth
17 \Rcolwidth=0.45\textwidth
18
```

12.1 Messages

All the error and warning messages are collected here as macros.

```
\eledpar@error
```

```
19 \newcommand{\eledpar@error}[2]{\PackageError{eledpar}{#1}{#2}}
```

```
\led@err@TooManyPstarts
```

```
20 \newcommand*{\led@err@TooManyPstarts}{%
21 \eledpar@error{Too many \string\pstart\space without printing.
22 Some text will be lost}{\@ehc}}
```

```
\led@err@BadLeftRightPstarts
```

```
23 \newcommand*{\led@err@BadLeftRightPstarts}[2]{%
24 \eledpar@error{The numbers of left (#1) and right (#2)
25 \string\pstart s do not match}{\@ehc}}
```

```
\led@err@LeftOnRightPage
```

```
\led@err@RightOnLeftPage 26 \newcommand*{\led@err@LeftOnRightPage}{%
27 \eledpar@error{The left page has ended on a right page}{\@ehc}}
28 \newcommand*{\led@err@RightOnLeftPage}{%
29 \eledpar@error{The right page has ended on a left page}{\@ehc}}
```

13 Sectioning commands

`\section@numR` This is the right side equivalent of `\section@num`.

Each section will read and write an associated ‘line-list file’, containing information used to do the numbering. Normally the file will be called `\jobname\.``nn`, where `nn` is the section number. However, for right side texts the file is called `\jobname\.``nnR`. The `\extensionchars` applies to the right side files just as it does to the normal files.

```
30 \newcount\section@numR
31 \section@numR=\z@
```

`\ifpst@rtedL` `\ifpst@rtedL` is set FALSE at the start of left side numbering, and similarly for `\ifpst@rtedR`. `\ifpst@rtedL` is defined in `eledmac`.

```
32 \pst@rtedLfalse
33 \newif\ifpst@rtedR
34 \pst@rtedRfalse
35
```

`\beginnumberingR` This is the right text equivalent of `\beginnumbering`, and begins a section of numbered text.

```
36 \newcommand*\beginnumberingR{%
37   \ifnumberingR
38     \led@err@NumberingStarted
39     \endnumberingR
40   \fi
41   \global\l@dnumpstartsR \z@
42   \global\pst@rtedRfalse
43   \global\numberingRtrue
44   \global\advance\section@numR \@ne
45   \global\absline@numR \z@
46   \gdef\normal@page@breakR{}
47   \gdef\l@prev@pbR{}
48   \gdef\l@prev@nopbR{}
49   \global\line@numR \z@
50   \global\@lockR \z@
51   \global\sub@lockR \z@
52   \global\sublines@false
53   \global\let\next@page@numR\relax
54   \global\let\sub@change\relax
55   \message{Section \the\section@numR R }%
56   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
57   \l@dend@stuff
58   \setcounter{pstartR}{1}
59   \begingroup
60   \initnumbering@sectcountR
61   \gdef\eled@sectionsR@{}%
62   \if@noeled@sec\else%
63     \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@numR R}{-}{-}\makeatother%
```

```

64 \immediate\openout\eled@sectioningR@out=\jobname.eledsec\the\section@numR R\relax%
65 \fi%
66 }

```

`\endnumbering` This is the left text version of the regular `\endnumbering` and must follow the last text for a left text numbered section. It sets `\ifpst@rtedL` to FALSE. It is fully defined in `eledmac`.

`\endnumberingR` This is the right text equivalent of `\endnumbering` and must follow the last text for a right text numbered section.

```

67 \def\endnumberingR{%
68 \ifnumberingR
69 \global\ numberingRfalse
70 \normal@pars
71 \ifl@dpairing
72 \global\pst@rtedRfalse
73 \else
74 \ifx\insertlines@listR\empty\else
75 \global\noteschanged@true
76 \fi
77 \ifx\line@listR\empty\else
78 \global\noteschanged@true
79 \fi
80 \fi
81 \ifnoteschanged@
82 \led@mess@NotesChanged
83 \fi
84 \else
85 \led@err@NumberingNotStarted
86 \fi
87 \endgroup
88 \if@noeled@sec\else%
89 \immediate\closeout\eled@sectioningR@out%
90 \fi%
91 }
92

```

`\initnumbering@sectcountR` We don't want the numbering of the right-side section commands to be continuous with the numbering of the left side, we switch the \LaTeX counter in `\numberingR`.

```

93 \newcounter{chapterR}
94 \newcounter{sectionR}
95 \newcounter{subsectionR}
96 \newcounter{subsubsectionR}
97 \newcommand{\initnumbering@sectcountR}{
98 \let\c@chapter\c@chapterR
99 \let\c@section\c@sectionR
100 \let\c@subsection\c@subsectionR
101 \let\c@subsubsection\c@subsubsectionR
102 }

```

`\pausenumberingR` These are the right text equivalents of `\pausenumbering` and `\resumenumbering`.
`\resumenumberingR`

```

103 \newcommand*{\pausenumberingR}{%
104   \endnumberingR\global\numberingRtrue}
105 \newcommand*{\resumenumberingR}{%
106   \ifnumberingR
107     \global\pst@rtedRtrue
108     \global\advance\section@numR \@ne
109     \led@mess@SectionContinued{\the\section@numR R}%
110     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
111     \l@dend@stuff
112     \begingroup%
113     \initnumbering@sectcountR%
114   \else
115     \led@err@numberingShouldHaveStarted
116     \endnumberingR
117     \beginnumberingR
118   \fi}
119
```

`\memorydumpL` `\memorydump` is a shorthand for `\pausenumbering\resumenumbering`. This will
`\memorydumpR` clear the memorised stuff for the previous chunks while keeping the numbering
going.

```

120 \newcommand*{\memorydumpL}{%
121   \endnumbering
122   \numberingtrue
123   \global\pst@rtedLtrue
124   \global\advance\section@num \@ne
125   \led@mess@SectionContinued{\the\section@num}%
126   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
127   \l@dend@stuff}
128 \newcommand*{\memorydumpR}{%
129   \endnumberingR
130   \numberingRtrue
131   \global\pst@rtedRtrue
132   \global\advance\section@numR \@ne
133   \led@mess@SectionContinued{\the\section@numR R}%
134   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
135   \l@dend@stuff}
136
```

14 Line counting

14.1 Choosing the system of lineation

Sometimes you want line numbers that start at 1 at the top of each page; sometimes you want line numbers that start at 1 at each `\pstart`; other times you want line

numbers that start at 1 at the start of each section and increase regardless of page breaks. `eledpar` lets you choose different schemes for the left and right texts.

`\ifbypstart@R` The `\ifbypage@R` and `\ifbypstart@R` flag specify the current lineation system:

`\bypstart@Rtrue` • line-of-page : `bypstart@R = false` and `bypage@R = true`.

`\bypstart@Rfalse` • line-of-pstart : `bypstart@R = true` and `bypage@R = false`.

`\ifbypage@R` `eledpar` will use the line-of-section system unless instructed otherwise.

`\bypage@Rtrue`

`\bypage@Rfalse`

```

137 \newif\ifbypage@R
138 \newif\ifbypstart@R
139   \bypage@Rfalse
140   \bypstart@Rfalse

```

`\lineationR` `\lineationR{<word>}` is the macro used to select the lineation system for right texts. Its argument is a string: either `page`, `pstart` or `section`.

```

141 \newcommand*{\lineationR}[1]{%
142   \ifnumbering
143     \led@err@LineationInNumbered
144   \else
145     \def\@tempa{#1}\def\@tempb{page}%
146     \ifx\@tempa\@tempb
147       \global\bypage@Rtrue
148       \global\bypstart@Rfalse
149     \else
150       \def\@tempb{pstart}%
151       \ifx\@tempa\@tempb
152         \global\bypage@Rfalse
153         \global\bypstart@Rtrue
154       \else
155         \def\@tempb{section}
156         \ifx\@tempa\@tempb
157           \global\bypage@Rfalse
158           \global\bypstart@Rfalse
159         \else
160           \led@warn@BadLineation
161         \fi
162       \fi
163     \fi
164   \fi}}

```

`\lineation*` `\lineation*` change the lineation system for the side.

```

165 \WithSuffix\newcommand\lineation*[1]{%
166   \lineation{#1}%
167   \lineationR{#1}%
168 }%

```

`\linenummargin` You call `\linenummargin{<word>}` to specify which margin you want your right

`\line@marginR` text's line numbers in; it takes one argument, a string. You can put the line

numbers in the same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. You can change this within a numbered section, but the change may not take effect just when you'd like; if it's done between paragraphs nothing surprising should happen.

For right texts the selection is recorded in the count `\line@marginR`, otherwise in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.

```

169 \newcount\line@marginR
170 \renewcommand*{\linenummargin}[1]{%
171   \l@getline@margin{#1}%
172   \ifnum\l@dttempcntb>\m@ne
173     \ifledRcol
174       \global\line@marginR=\l@dttempcntb
175     \else
176       \global\line@margin=\l@dttempcntb
177     \fi
178   \fi}}

```

By default put right text numbers at the right.

```

179 \line@marginR=\@ne
180

```

`\c@firstlinenumR` The following counters tell `eledmac` which right text lines should be printed with
`\c@linenumincrementR` line numbers. `firstlinenum` is the number of the first line in each section that gets a number; `linenumincrement` is the difference between successive numbered lines. The initial values of these counters produce labels on lines 5, 10, 15, etc. `linenumincrement` must be at least 1.

```

181 \newcounter{firstlinenumR}
182 \setcounter{firstlinenumR}{5}
183 \newcounter{linenumincrementR}
184 \setcounter{linenumincrementR}{5}

```

`\c@firstsublinenumR` The following parameters are just like `firstlinenumR` and `linenumincrementR`,
`\c@sublinenumincrementR` but for sub-line numbers. `sublinenumincrementR` must be at least 1.

```

185 \newcounter{firstsublinenumR}
186 \setcounter{firstsublinenumR}{5}
187 \newcounter{sublinenumincrementR}
188 \setcounter{sublinenumincrementR}{5}
189

```

`\firstlinenum` These are the user's macros for changing (sub) line numbers. They are defined in
`\linenumincrement` `eledmac v0.7`, but just in case I have started by `\provideing` them. The starred
`\firstsublinenum` versions are specific to `eledpar`.
`\sublinenumincrement` 190 `\providecommand*{\firstlinenum}{}`
`\firstlinenum*` 191 `\providecommand*{\linenumincrement}{}`
`\linenumincrement*` 192 `\providecommand*{\firstsublinenum}{}`
`\firstsublinenum*` 193 `\providecommand*{\sublinenumincrement}{}`
`\sublinenumincrement*` 194 `\renewcommand*{\firstlinenum}[1]{%`
 195 `\ifledRcol \setcounter{firstlinenumR}{#1}%`

```

196 \else      \setcounter{firstlinenum}{#1}%
197 \fi}
198 \renewcommand*{\linenumincrement}[1]{%
199 \ifledRcol \setcounter{linenumincrementR}{#1}%
200 \else      \setcounter{linenumincrement}{#1}%
201 \fi}
202 \renewcommand*{\firstsublinenum}[1]{%
203 \ifledRcol \setcounter{firstsublinenumR}{#1}%
204 \else      \setcounter{firstsublinenum}{#1}%
205 \fi}
206 \renewcommand*{\sublinenumincrement}[1]{%
207 \ifledRcol \setcounter{sublinenumincrementR}{#1}%
208 \else      \setcounter{sublinenumincrement}{#1}%
209 \fi}
210 \WithSuffix\newcommand\firstlinenum*[1]{\setcounter{firstlinenumR}{#1}\setcounter{firstlinenum}{#1}}
211 \WithSuffix\newcommand\linenumincrement*[1]{\setcounter{linenumincrementR}{#1}\setcounter{linenumincr
212 \WithSuffix\newcommand\firstsublinenum*[1]{\setcounter{subfirstlinenumR}{#1}\setcounter{subfirstlinen
213 \WithSuffix\newcommand\sublinenumincrement*[1]{\setcounter{sublinenumincrementR}{#1}\setcounter{subli

```

`\Rlineflag` This is appended to the line numbers of right text.

```

214 \newcommand*{\Rlineflag}{R}
215

```

`\linenumrepR` `\linenumrepR{<ctr>}` typesets the right line number `<ctr>`, and similarly `\sublinenumrepR` for subline numbers.

```

216 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
217 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
218

```

`\leftlinenumR` `\leftlinenumR` and `\rightlinenumR` are the macros that are called to print the right text's marginal line numbers. Much of the code for these is common and is maintained in `\l@dlinenumR`.

```

219 \newcommand*{\leftlinenumR}{%
220 \l@dlinenumR
221 \kern\linenumsep}
222 \newcommand*{\rightlinenumR}{%
223 \kern\linenumsep
224 \l@dlinenumR}
225 \newcommand*{\l@dlinenumR}{%
226 \numlabfont\linenumrepR{\line@numR}\Rlineflag%
227 \ifsublines@
228 \ifnum\subline@num>\z@
229 \unskip\fullstop\sublinenumrepR{\subline@numR}%
230 \fi
231 \fi}
232

```

14.2 Line-number counters and lists

We need another set of counters and lists for the right text, corresponding to those in `eledmac` for regular or left text.

`\line@numR` The count `\line@numR` stores the line number that's used in the right text's marginal line numbering and in notes. The count `\subline@numR` stores a sub-line number that qualifies `\line@numR`. The count `\absline@numR` stores the absolute number of lines since the start of the right text section: that is, the number we've actually printed, no matter what numbers we attached to them.

```
233 \newcount\line@numR
234 \newcount\subline@numR
235 \newcount\absline@numR
236
```

`\line@listR` Now we can define the list macros that will be created from the line-list file. They are directly analogous to the left text ones. The full list of action codes and their meanings is given in the `eledmac` manual.

`\actions@listR` Here are the commands to create these lists:

```
237 \list@create{\line@listR}
238 \list@create{\insertlines@listR}
239 \list@create{\actionlines@listR}
240 \list@create{\actions@listR}
241 \list@create{\sw@listR}%
242 \list@create{\sw@list@inedtextR}%
243
```

`\linesinpar@listL` In order to synchronise left and right chunks in parallel processing we need to know how many lines are in each left and right text chunk, and the maximum of these for each pair of chunks.

`\linesinpar@listR`
`\maxlinesinpar@list`

```
244 \list@create{\linesinpar@listL}
245 \list@create{\linesinpar@listR}
246 \list@create{\maxlinesinpar@list}
247
```

`\page@numR` The right text page number.

```
248 \newcount\page@numR
249
```

14.3 Reading the line-list file

`\read@linelist` `\read@linelist{<file>}` is the control sequence that's called by `\beginnumbering` (via `\line@list@stuff`) to open and process a line-list file; its argument is the name of the file.

```
250 \renewcommand*\read@linelist[1]{%
```


We do do different things depending whether or not we are processing right text

```

251 \ifledRcol
252   \list@clear{\line@listR}%
253   \list@clear{\insertlines@listR}%
254   \list@clear{\actionlines@listR}%
255   \list@clear{\actions@listR}%
256   \list@clear{\linesinpar@listR}%
257   \list@clear{\linesonpage@listR}%
258   \list@clear{\sw@listR}%
259   \list@clear{\sw@list@inedtextR}%
260 \else
261   \list@clearing@reg
262   \list@clear{\linesinpar@listL}%
263   \list@clear{\linesonpage@listL}%
264 \fi

```

Make sure that the `\maxlinesinpar@list` is empty (otherwise things will be thrown out of kilter if there is any old stuff still hanging in there).

```

265 \list@clear{\maxlinesinpar@list}

```

Now get the file and interpret it.

```

266 \get@linelistfile{#1}%
267 \endgroup

```

When the reading is done, we're all through with the line-list file. All the information we needed from it will now be encoded in our list macros. Finally, we initialize the `\next@actionline` and `\next@action` macros, which specify where and what the next action to be taken is.

```

268 \ifledRcol
269   \global\page@numR=\m@ne
270   \ifx\actionlines@listR\empty
271     \gdef\next@actionlineR{1000000}%
272   \else
273     \gl@p\actionlines@listR\to\next@actionlineR
274     \gl@p\actions@listR\to\next@actionR
275   \fi
276 \else
277   \global\page@num=\m@ne
278   \ifx\actionlines@list\empty
279     \gdef\next@actionline{1000000}%
280   \else
281     \gl@p\actionlines@list\to\next@actionline
282     \gl@p\actions@list\to\next@action
283   \fi
284 \fi}
285

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. The `\memorydump` macro is available if you run into macro memory limitations.

14.4 Commands within the line-list file

This section defines the commands that can appear within a line-list file, except for `\@lab` which is in a later section among the cross-referencing commands it is associated with.

The macros with `action` in their names contain all the code that modifies the action-code list.

`\@nl@regR` `\@nl` does everything related to the start of a new line of numbered text. Exactly what it does depends on whether right text is being processed.

```

286 \newcommand{\@nl@regR}{%
287   \ifx\l@dchset@num\relax \else
288     \advance\absline@numR \@ne
289     \set@line@action
290     \let\l@dchset@num\relax
291     \advance\absline@numR \m@ne
292     \advance\line@numR \m@ne%    % do we need this?
293   \fi
294   \advance\absline@numR \@ne
295   \ifx\next@page@numR\relax \else
296     \page@action
297     \let\next@page@numR\relax
298   \fi
299   \ifx\sub@change\relax \else
300     \ifnum\sub@change>\z@
301       \sublines@true
302     \else
303       \sublines@false
304     \fi
305     \sub@action
306     \let\sub@change\relax
307   \fi
308   \ifcase\@lockR
309   \or
310     \@lockR \tw@
311   \or\or
312     \@lockR \z@
313   \fi
314   \ifcase\sub@lockR
315   \or
316     \sub@lockR \tw@
317   \or\or
318     \sub@lockR \z@
319   \fi
320   \ifsublines@
321     \ifnum\sub@lockR<\tw@
322       \advance\subline@numR \@ne
323     \fi
324   \else
325     \ifnum\@lockR<\tw@

```

```

326     \advance\line@numR \@ne \subline@numR \z@
327     \fi
328 \fi}
329
330 \renewcommand*{\@nl}[2]{%
331   \fix@page{#1}%
332   \ifledRcol
333     \@nl@regR
334   \else
335     \@nl@reg
336   \fi}
337

```

\last@page@numR We have to adjust \fix@page to handle parallel texts.

```

\fix@page
338 \newcount\last@page@numR
339 \last@page@numR=-10000
340 \renewcommand*{\fix@page}[1]{%
341   \ifledRcol
342     \ifnum #1=\last@page@numR
343     \else
344       \ifbypage@R
345         \line@numR \z@ \subline@numR \z@
346       \fi
347       \page@numR=#1\relax
348       \last@page@numR=#1\relax
349       \def\next@page@numR{#1}%
350     \fi
351   \else
352     \ifnum #1=\last@page@num
353     \else
354       \ifbypage@
355         \line@num \z@ \subline@num \z@
356       \fi
357       \page@num=#1\relax
358       \last@page@num=#1\relax
359       \def\next@page@num{#1}%
360       \listxadd{\normal@page@break}{\the\absline@num}
361     \fi
362   \fi}
363

```

\@adv The \@adv{<num>} macro advances the current visible line number by the amount specified as its argument. This is used to implement \advanceline.

```

364 \renewcommand*{\@adv}[1]{%
365   \ifsublines@
366     \ifledRcol
367       \advance\subline@numR by #1\relax
368       \ifnum\subline@numR<\z@
369         \led@warn@BadAdvancelineSubline

```

```

370     \subline@numR \z@
371     \fi
372   \else
373     \advance\subline@num by #1\relax
374     \ifnum\subline@num<\z@
375       \led@warn@BadAdvancelineSubline
376       \subline@num \z@
377     \fi
378   \fi
379 \else
380   \iflabeledRcol
381     \advance\line@numR by #1\relax
382     \ifnum\line@numR<\z@
383       \led@warn@BadAdvancelineLine
384       \line@numR \z@
385     \fi
386   \else
387     \advance\line@num by #1\relax
388     \ifnum\line@num<\z@
389       \led@warn@BadAdvancelineLine
390       \line@num \z@
391     \fi
392   \fi
393 \fi
394 \set@line@action}
395

```

`\@set` The `\@set{<num>}` macro sets the current visible line number to the value specified as its argument. This is used to implement `\setline`.

```

396 \renewcommand*{\@set}[1]{%
397   \iflabeledRcol
398     \ifsublines@
399       \subline@numR=#1\relax
400     \else
401       \line@numR=#1\relax
402     \fi
403     \set@line@action
404   \else
405     \ifsublines@
406       \subline@num=#1\relax
407     \else
408       \line@num=#1\relax
409     \fi
410     \set@line@action
411   \fi}
412

```

`\l@d@set` The `\l@d@set{<num>}` macro sets the line number for the next `\pstart...` to the value specified as its argument. This is used to implement `\setlinenum`.

`\l@dchset@num` is a flag to the `\@l` macro. If it is not `\relax` then a linenumber change is to be done.

```

413 \renewcommand*{\l@d@set}[1]{%
414   \ifledRcol
415     \line@numR=#1\relax
416     \advance\line@numR \@ne
417     \def\l@dchset@num{#1}
418   \else
419     \line@num=#1\relax
420     \advance\line@num \@ne
421     \def\l@dchset@num{#1}
422   \fi}
423 \let\l@dchset@num\relax
424

```

`\page@action` `\page@action` adds an entry to the action-code list to change the page number.

```

425 \renewcommand*{\page@action}{%
426   \ifledRcol
427     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
428     \xright@appenditem{\next@page@numR}\to\actions@listR
429   \else
430     \xright@appenditem{\the\absline@num}\to\actionlines@list
431     \xright@appenditem{\next@page@num}\to\actions@list
432   \fi}

```

`\set@line@action` `\set@line@action` adds an entry to the action-code list to change the visible line number.

```

433 \renewcommand*{\set@line@action}{%
434   \ifledRcol
435     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
436     \ifsublines@
437       \@l@dttempcnta=-\subline@numR
438     \else
439       \@l@dttempcnta=-\line@numR
440     \fi
441     \advance\@l@dttempcnta by -5000\relax
442     \xright@appenditem{\the\@l@dttempcnta}\to\actions@listR
443   \else
444     \xright@appenditem{\the\absline@num}\to\actionlines@list
445     \ifsublines@
446       \@l@dttempcnta=-\subline@num
447     \else
448       \@l@dttempcnta=-\line@num
449     \fi
450     \advance\@l@dttempcnta by -5000\relax
451     \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
452   \fi}
453

```

`\sub@action` `\sub@action` adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the `\ifsublines@` flag.

```

454 \renewcommand*{\sub@action}{%
455   \ifledRcol
456     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
457     \ifsublines@
458       \xright@appenditem{-1001}\to\actions@listR
459     \else
460       \xright@appenditem{-1002}\to\actions@listR
461     \fi
462   \else
463     \xright@appenditem{\the\absline@num}\to\actionlines@list
464     \ifsublines@
465       \xright@appenditem{-1001}\to\actions@list
466     \else
467       \xright@appenditem{-1002}\to\actions@list
468     \fi
469   \fi}
470

```

`\do@lockon` `\lock@on` adds an entry to the action-code list to turn line number locking on.
`\do@lockonR` The current setting of the sub-lineation flag tells us whether this applies to line numbers or sub-line numbers.

```

471 \newcount\@lockR
472 \newcount\sub@lockR
473
474 \newcommand*{\do@lockonR}{%
475   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
476   \ifsublines@
477     \xright@appenditem{-1005}\to\actions@listR
478     \ifnum\sub@lockR=\z@
479       \sub@lockR \@ne
480     \else
481       \ifnum\sub@lockR=\thr@@
482         \sub@lockR \@ne
483       \fi
484     \fi
485   \else
486     \xright@appenditem{-1003}\to\actions@listR
487     \ifnum\@lockR=\z@
488       \@lockR \@ne
489     \else
490       \ifnum\@lockR=\thr@@
491         \@lockR \@ne
492       \fi
493     \fi
494   \fi}
495
496 \renewcommand*{\do@lockon}{%

```

```

497 \ifx\next\lock@off
498   \global\let\lock@off=\skip@lockoff
499 \else
500   \ifledRcol
501     \do@lockonR
502   \else
503     \do@lockonL
504   \fi
505 \fi}

```

`\lock@off` `\lock@off` adds an entry to the action-code list to turn line number locking off.

```

\do@lockoff 506
\do@lockoffR 507
\skip@lockoff 508 \newcommand{\do@lockoffR}{%
509   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
510   \ifsublines@
511     \xright@appenditem{-1006}\to\actions@listR
512     \ifnum\sub@lockR=\tw@
513       \sub@lockR \thr@@
514     \else
515       \sub@lockR \z@
516     \fi
517   \else
518     \xright@appenditem{-1004}\to\actions@listR
519     \ifnum\@lockR=\tw@
520       \@lockR \thr@@
521     \else
522       \@lockR \z@
523     \fi
524   \fi}
525
526 \renewcommand*{\do@lockoff}{%
527   \ifledRcol
528     \do@lockoffR
529   \else
530     \do@lockoffL
531   \fi}
532 \global\let\lock@off=\do@lockoff
533

```

`\n@num` This macro implements the `\skipnumbering` command. It uses a new action code, namely 1007.

```

534 \providecommand*\n@num{}
535 \renewcommand*\n@num{}
536 \ifledRcol
537   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
538   \xright@appenditem{-1007}\to\actions@listR
539 \else
540   \n@num@reg
541 \fi}

```

542

`\@ref` marks the start of a passage, for creation of a footnote reference. It takes two arguments:

- #1, the number of entries to add to `\insertlines@list` for this reference. This value for right text, here and within `\edtext`, which computes it and writes it to the line-list file, will be stored in the count `\insert@countR`.

543 `\newcount\insert@countR`

- #2, a sequence of other line-list-file commands, executed to determine the ending line-number. (This may also include other `\@ref` commands, corresponding to uses of `\edtext` within the first argument of another instance of `\edtext`.)

The first thing `\@ref` itself does is to add the specified number of items to the `\insertlines@list` list.

```
544 \renewcommand*{\@ref}[2]{%
545   \ifledRcol
546     \global\insert@countR=#1\relax
547     \loop\ifnum\insert@countR>\z@
548       \xright@appenditem{\the\absline@numR}\to\insertlines@listR
549       \global\advance\insert@countR \m@ne
550     \repeat
```

Next, process the second argument to determine the page and line numbers for the end of this lemma. We temporarily equate `\@ref` to a different macro that just executes its argument, so that nested `\@ref` commands are just skipped this time. Some other macros need to be temporarily redefined to suppress their action.

```
551 \begingroup
552   \let\@ref=\dummy@ref
553   \let\@lopR\@gobble
554   \let\page@action=\relax
555   \let\sub@action=\relax
556   \let\set@line@action=\relax
557   \let\@lab=\relax
558   \let\@sw\@gobbletwo%
559   #2
560   \global\endpage@num=\page@numR
561   \global\endline@num=\line@numR
562   \global\endsubline@num=\subline@numR
563 \endgroup
```

Now store all the information about the location of the lemma's start and end in `\line@list`.

```
564   \xright@appenditem%
565     {\the\page@numR|\the\line@numR}%
566     \ifsublines@ \the\subline@numR \else 0\fi}%
567     \the\endpage@num|\the\endline@num}%
568     \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
```


Finally, execute the second argument of `\@ref` again, to perform for real all the commands within it.

```
569 #2
570 \else
```

And when not in right text

```
571 \@ref@reg{#1}{#2}%
572 \fi}
```

`\@pend` `\@pend{<num>}` adds its argument to the `\linesinpar@listL` list, and analogously `\@pendR` for `\@pendR`. If needed, it resets line number. We start off with a `\providecommand` just in case an older version of `eledmac` is being used which does not define these macros.

```
573 \providecommand*{\@pend}[1]{%
574 \renewcommand*{\@pend}[1]{%
575 \ifbypstart@global\line@num=0\fi%
576 \xright@appenditem{#1}\to\linesinpar@listL}
577 \providecommand*{\@pendR}[1]{%
578 \renewcommand*{\@pendR}[1]{%
579 \ifbypstart@R\global\line@numR=0\fi
580 \xright@appenditem{#1}\to\linesinpar@listR}
581
```

`\@lopL` `\@lopL{<num>}` adds its argument to the `\linesonpage@listL` list, and analogously `\@lopR` for `\@lopR`. We start off with a `\providecommand` just in case an older version of `eledmac` is being used which does not define these macros.

```
582 \providecommand*{\@lopL}[1]{%
583 \renewcommand*{\@lopL}[1]{%
584 \xright@appenditem{#1}\to\linesonpage@listL}
585 \providecommand*{\@lopR}[1]{%
586 \renewcommand*{\@lopR}[1]{%
587 \xright@appenditem{#1}\to\linesonpage@listR}
588
```

14.5 Writing to the line-list file

We've now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we'll cover the commands that `eledmac` uses within the text of a section to write commands out to the line-list.

`\linenum@outR` The file for right texts will be opened on output stream `\linenum@outR`.

```
589 \newwrite\linenum@outR
```

`\iffirst@linenum@out@R` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open.

```
\first@linenum@out@Rtrue
\first@linenum@out@Rfalse 590 \newif\iffirst@linenum@out@R
591 \first@linenum@out@Rtrue
```

`\line@list@stuffR` This is the right text version of the `\line@list@stuff{<file>}` macro. It is called by `\beginnumberingR` and performs all the line-list operations needed at the start of a section. Its argument is the name of the line-list file.

```

592 \newcommand*{\line@list@stuffR}[1]{%
593   \read@linelist{#1}%
594   \iffirst@linenum@out@R
595     \immediate\closeout\linenum@outR
596     \global\first@linenum@out@Rfalse
597     \immediate\openout\linenum@outR=#1
598   \else
599     \if@minipage%
600       \leavevmode%
601     \fi%
602     \closeout\linenum@outR%
603     \openout\linenum@outR=#1%
604   \fi}
605

```

`\new@lineL` The `\new@lineL` macro sends the `\@nl` command to the left text line-list file, to mark the start of a new text line.

```

606 \newcommand*{\new@lineL}{%
607   \write\linenum@out{\string\@nl[\the\c@page][\thepage]}}

```

`\new@lineR` The `\new@lineR` macro sends the `\@nl` command to the right text line-list file, to mark the start of a new text line.

```

608 \newcommand*{\new@lineR}{%
609   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}

```

`\flag@start` We enclose a lemma marked by `\edtext` in `\flag@start` and `\flag@end`: these `\flag@end` send the `\@ref` command to the line-list file.

`\startsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate `\endsub` instructions to the line-list file.

```

610 \renewcommand*{\startsub}{\dimen0\lastskip
611   \ifdim\dimen0>0pt \unskip \fi
612   \ifledRcol \write\linenum@outR{\string\sub@on}%
613   \else      \write\linenum@out{\string\sub@on}%
614   \fi
615   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
616 \def\endsub{\dimen0\lastskip
617   \ifdim\dimen0>0pt \unskip \fi
618   \ifledRcol \write\linenum@outR{\string\sub@off}%
619   \else      \write\linenum@out{\string\sub@off}%
620   \fi
621   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
622

```

`\advanceline` You can use `\advanceline{<num>}` in running text to advance the current visible line-number by a specified value, positive or negative.

```

623 \renewcommand*{\advanceline}[1]{%
624   \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
625   \else      \write\linenum@out{\string\@adv[#1]}%
626   \fi}

```

\setline You can use `\setline{<num>}` in running text (i.e., within `\pstart... \pend`) to set the current visible line-number to a specified positive value.

```

627 \renewcommand*{\setline}[1]{%
628   \ifnum#1<\z@
629     \led@warn@BadSetline
630   \else
631     \ifledRcol \write\linenum@outR{\string\@set[#1]}%
632     \else      \write\linenum@out{\string\@set[#1]}%
633     \fi
634   \fi}

```

\setlinenum You can use `\setlinenum{<num>}` before a `\pstart` to set the visible line-number to a specified positive value. It writes a `\l@d@set` command to the line-list file.

```

635 \renewcommand*{\setlinenum}[1]{%
636   \ifnum#1<\z@
637     \led@warn@BadSetlinenum
638   \else
639     \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
640     \else      \write\linenum@out{\string\l@d@set[#1]} \fi
641   \fi}
642

```

\startlock You can use `\startlock` or `\endlock` in running text to start or end line number locking at the current line. They decide whether line numbers or sub-line numbers are affected, depending on the current state of the sub-lineation flags.

```

643 \renewcommand*{\startlock}{%
644   \ifledRcol \write\linenum@outR{\string\lock@on}%
645   \else      \write\linenum@out{\string\lock@on}%
646   \fi}
647 \def\endlock{%
648   \ifledRcol \write\linenum@outR{\string\lock@off}%
649   \else      \write\linenum@out{\string\lock@off}%
650   \fi}
651

```

\skipnumbering In numbered text, `\skipnumbering` in a line will suspend the numbering for that particular line. That is, line numbers are unchanged and no line number will be printed.

```

652 \renewcommand*{\skipnumbering}{%
653   \ifledRcol \write\linenum@outR{\string\n@num}%
654   \advanceline{-1}%
655   \else
656     \skipnumbering@reg
657   \fi}
658

```

15 Marking text for notes

The `\edtext` (or `\critext`) macro is used to create all footnotes and endnotes, as well as to print the portion of the main text to which a given note or notes is keyed. The idea is to have that lemma appear only once in the `.tex` file: all instances of it in the main text and in the notes are copied from that one appearance.

`\critext` requires two arguments. At any point within numbered text, you use it by saying:

```
\critext{#1}#2/
```

Similarly `\edtext` requires the same two arguments but you use it by saying:

```
\edtext{#1}{#2}
```

```
\critext And similarly for \edtext.
\edtext
\set@line The \set@line macro is called by \edtext to put the line-reference field and font
specifier for the current block of text into \l@d@nums.
659 \renewcommand*{\set@line}{%
660   \ifledRcol
661     \ifx\line@listR\empty
662       \global\noteschanged@true
663       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
664     \else
665       \gl@p\line@listR\to\@tempb
666       \xdef\l@d@nums{\@tempb|\edfont@info}%
667       \global\let\@tempb=\undefined
668     \fi
669   \else
670     \ifx\line@list\empty
671       \global\noteschanged@true
672       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
673     \else
674       \gl@p\line@list\to\@tempb
675       \xdef\l@d@nums{\@tempb|\edfont@info}%
676       \global\let\@tempb=\undefined
677     \fi
678   \fi}
679
```

15.1 Specific hooks and commands for notes

The `\eledmac \newseries@` initializes commands which are linked to notes series. However, to keep `eledmac` as light as possible, it does not define commands which are specific to `eledpar`. This is what does `\newseries@eledpar`. The specific hooks are also defined here.

`\newseries@eledpar`

```
680 \newcommand{\newseries@eledpar}[1]{%
```

15.1.1 Create commands to memorize display options

```
681 \csgdef{onlysideX@#1}{}%
```

```
682 \csgdef{onlyXside@#1}{}%
```

15.1.2 Boolean flags for notes to be printed in one side only

`eledpar` allows notes to be printed on one side only. We need boolean flags, and set them to true when a note series is not printed in one side even though it contains something.

```
683 \global\newbool{keepforXside@#1}%
```

```
684 \global\newbool{keepforsideX@#1}%
```

15.1.3 Familiar footnotes without marks

The `\footnoteXnomk` commands are for notes which are printed in the left side, while they are called in the right side. Basically, they set first toggle `\nomark@` to true, then call the `\footnoteX`. and finally add the footnote counter in the footnote counter list.

So declare the list.

```
685 \expandafter\list@create\csname footnote#1@mk\endcsname%
```

Then, declare the `\footnoteXnomk` command.

```
686 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
```

First step: just call the normal `\footnoteX`, saying that we don't want to print the mark.

```
687 \toggletrue{nomk@}%
```

```
688 \csuse{footnote#1}{##1}%
```

```
689 \togglefalse{nomk@}%
```

Second, and last, step: store the footnote counter in the footnote counters list. We use some `\let`, because `\xright@appenditem` is difficult to use with `\expandafter`.

```
690 \letcs{\@tmp}{footnote#1@mk}%
```

```
691 \numdef\@tmpa{\csuse{c@footnote#1}}%
```

```
692 \global\xright@appenditem{\@tmpa}\to\@tmp%
```

```
693 \global\cslet{footnote#1@mk}{\@tmp}%
```

```
694 }%
```

Then, declare the command which inserts the footnotemark in the right side.

```
695 \expandafter\newcommand\csname footnote#1mk\endcsname{%
```

Get the first element of the footnote mark list. As `\gl@p` is difficult to use with dynamic name macro, we use `\let` commands.

```
696 \letcs{\@tmp}{footnote#1@mk}%
```

```
697 \gl@p\@tmp\to\@tmpa%
```

```
698 \global\cslet{footnote#1@mk}{\@tmp}%
```

Set the footnotecounter with it. For the sake of security, we make a backup of the previous value.

```

699      \letcs{\old@footnote}{c@footnote#1}%
700      \setcounter{footnote#1}{\@tmpa}%

Define the footnote mark and print it
701      \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
702      \csuse{@footnotemark#1}%

Restore previous footnote counter and finally add space.
703      \setcounter{footnote#1}{\old@footnote}%
704      \xspace%
705  }%

End of \newseries@eledpar.
706 }%
```

15.1.4 Create hooks

Read the eledmac code handbook about `\newhookcommand@series`. Here, we create hooks which are specific to eledpar.

```

707 \newhookcommand@series{onlysideX}%
708
709 \newhookcommand@series{onlyXside}%
710
```

15.1.5 Init standards series (A,B,C,D,E,Z)

`\init@series@eledpar` `\newseries@eledpar` is called by `\newseries@`. However, this command is called before eledpar is loaded. Thus, we need to initiate a specific series hook for eledpar.

```

711 \newcommand{\init@series@eledpar}{%
712   \def\do##1{\newseries@eledpar{##1}}%
713   \dolistloop{\@series}%
714 }%
715 \init@series@eledpar%
```

16 Pstart numbers dumping and restoration

While in eledmac the footnotes are inserted in the same time as the `\pstart` ... `\pend` are read, in eledpar they are inserted when the `\Columns` or `\Pages` commands are called. Consequently, if we do nothing, the value of the `PstartL` and `PstartR` counters are not the same in the main text and in the notes. To solve this problem, we dump the values in two list (one by side) when processing `\pstart` and restore these at each `\pstart` when calling `\Columns` or `\Pages`. We also dump and restore the value of the boolean `\ifnumberpstart`.

So, first step, creating the lists. Here, “pc” means “public counters”.

```

\list@pstartL@pc
\list@pstartR@pc 716 \list@create{\list@pstartL@pc}%
                  717 \list@create{\list@pstartR@pc}%

```

Two commands to dump current pstarts. We prefer two commands to one with argument indicating the side, because the commands are short, and so we save one test (or a `\csname` construction).

```

\dump@pstartL@pc
\dump@pstartR@pc 718 \def\dump@pstartL@pc{%
                  719 \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
                  720 \global\cslet{numberpstart@L\the\l@dnumpstartsL}{\ifnumberpstart}%
                  721 }%
                  722
                  723 \def\dump@pstartR@pc{%
                  724 \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
                  725 \global\cslet{numberpstart@R\the\l@dnumpstartsR}{\ifnumberpstart}%
                  726 }%
                  727

```

```

\restore@pstartL@pc And so, the commands to restore them
\restore@pstartR@pc 728 \def\restore@pstartL@pc{%
                    729 \ifx\list@pstartL@pc\empty\else%
                    730 \gl@p\list@pstartL@pc\to\@temp%
                    731 \global\c@pstartL=\@temp%
                    732 \fi%
                    733 }%
                    734 \def\restore@pstartR@pc{%
                    735 \ifx\list@pstartR@pc\empty\else%
                    736 \gl@p\list@pstartR@pc\to\@temp%
                    737 \global\c@pstartR=\@temp%
                    738 \fi%
                    739 }%

```

17 Parallel environments

The initial set up for parallel processing is deceptively simple.

`pairs` The `pairs` environment is for parallel columns and the `pages` environment for parallel pages.

```

chapterinpages 740 \newenvironment{pairs}{%}
                741 \l@dpairingtrue
                742 \l@dpagingfalse
                743 \initnumbering@sectcmd
                744 \at@begin@pairs%
                745 }{%
                746 \l@dpairingfalse
                747 }
                748

```

`\AtBeginPairs` The `\AtBeginPairs` macro just define a `\at@begin@pairs` macro, called at the beginning of each `pairs` environments.

```
749 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
750 \def\at@begin@pairs{}%
751
```

The `pages` environment additionally sets the ‘column’ widths to the `\textwidth` (as known at the time the package is called). In this environment, there are two text in parallel on 2 pages. To prevent chapters starting on a lefthand page, the `\chapter` command is redefined to not clear pages.

```
752 \newenvironment{pages}{%
753   \let\oldchapter\chapter
754   \let\chapter\chapterinpages
755   \l@dpairingtrue
756   \l@dpagingtrue
757   \initnumbering@sectcmd
758   \setlength{\Lcolwidth}{\textwidth}%
759   \setlength{\Rcolwidth}{\textwidth}%
760 }{%
761   \l@dpairingfalse
762   \l@dpagingfalse
763   \let\chapter\oldchapter
764 }
765 \newcommand{\chapterinpages}{\thispagestyle{plain}%
766                               \global\@topnum\z@
767                               \@afterindentfalse
768                               \secdef\@chapter\@schapter}
769
```

`ifinstanzaL` These boolean tests are switched by the `\stanza` command, using either the left or right side.

```
770 \newif\ifinstanzaL
771 \newif\ifinstanzaR
```

`Leftside` Within the `pairs` and `pages` environments the left and right hand texts are within `Leftside` and `Rightside` environments, respectively. The `Leftside` environment is simple, indicating that right text is not within its purview and using some particular macros.

```
772 \newenvironment{Leftside}{%
773   \ledRcolfalse
774   \setcounter{pstartL}{1}
775   \let\pstart\pstartL
776   \let\thepstart\thepstartL
777   \let\pend\pendL
778   \let\memorydump\memorydumpL
779   \Leftsidehook
780   \let\old@startstanza\@startstanza
781   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
```



```

782 }{
783   \Leftsidehookend}

```

`\Leftsidehook` Hooks into the start and end of the `Leftside` and `Rightside` environments. These
`\Leftsidehookend` are initially empty.

```

\Rightsidehook 784 \newcommand*{\Leftsidehook}{}
\Rightsidehookend 785 \newcommand*{\Leftsidehookend}{}
                  786 \newcommand*{\Rightsidehook}{}
                  787 \newcommand*{\Rightsidehookend}{}
                  788

```

`Rightside` The `Rightside` environment is only slightly more complicated than the `Leftside`. Apart from indicating that right text is being provided it ensures that the right right text code will be used.

```

789 \newenvironment{Rightside}{%
790   \ledRcoltrue
791   \let\beginnumbering\beginnumberingR
792   \let\endnumbering\endnumberingR
793   \let\pausenumbering\pausenumberingR
794   \let\resumenumbering\resumenumberingR
795   \let\memorydump\memorydumpR
796   \let\thepstart\thepstartR
797   \let\pstart\pstartR
798   \let\pend\pendR
799   \let\ledpb\ledpbR
800   \let\lednopb\lednopbR
801   \let\lineation\lineationR
802   \Rightsidehook
803   \let\old@startstanza\@startstanza
804   \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
805 }{%
806   \ledRcolfalse
807   \Rightsidehookend
808 }
809

```

18 Paragraph decomposition and reassembly

In order to be able to count the lines of text and affix line numbers, we add an extra stage of processing for each paragraph. We send the paragraph into a box register, rather than straight onto the vertical list, and when the paragraph ends we slice the paragraph into its component lines; to each line we add any notes or line numbers, add a command to write to the line-list, and then at last send the line to the vertical list. This section contains all the code for this processing.

18.1 Boxes, counters, \pstart and \pend

`\num@linesR` Here are numbers and flags that are used internally in the course of the paragraph decomposition.

`\one@lineR`

`\par@lineR` When we first form the paragraph, it goes into a box register, `\l@dLcolrawbox` or `\l@dRcolrawbox` for right text, instead of onto the current vertical list. The `\ifnumberedpar@` flag will be `true` while a paragraph is being processed in that way. `\num@lines(R)` will store the number of lines in the paragraph when it's complete. When we chop it up into lines, each line in turn goes into the `\one@line` or `\one@lineR` register, and `\par@line(R)` will be the number of that line within the paragraph.

```
810 \newcount\num@linesR
```

```
811 \newbox\one@lineR
```

```
812 \newcount\par@lineR
```

`\pstartL` `\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant variables, and then arranges for the subsequent text to go into the appropriate box. `\pstart` needs to appear at the start of every paragraph that's to be numbered.

`\pstartR`

Beware: everything that occurs between `\pstart` and `\pend` is happening within a group; definitions must be global if you want them to survive past the end of the paragraph.

We have to have specific left and right `\pstart` when parallel processing; among other things because of potential changes in the linewidth.

```
813
```

```
814 \newcounter{pstartL}
```

```
815 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
```

```
816 \newcounter{pstartR}
```

```
817 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
```

```
818
```

```
819 \newcommandx*\pstartL}[1][1]{%
```

```
820   \if@nobreak%
```

```
821     \let\oldnobreak\@nobreaktrue%
```

```
822   \else%
```

```
823     \let\oldnobreak\@nobreakfalse%
```

```
824   \fi%
```

```
825     \@nobreaktrue%
```

```
826   \ifluatex%
```

```
827     \xdef\l@luatextextdir@L{\the\luatextextdir}%
```

```
828     \xdef\l@luatexpardir@L{\the\luatexpardir}%
```

```
829     \xdef\l@luatexbodydir@L{\the\luatexbodydir}%
```

```
830   \fi%
```

```
831   \ifnumbering \else%
```

```
832     \led@err@PstartNotNumbered%
```

```
833     \beginnumbering%
```

```
834   \fi%
```

```
835   \ifnumberedpar@%
```

```
836     \led@err@PstartInPstart%
```

```
837 \pend%
838 \fi%
```

If this is the first \pstart in a numbered section, clear any inserts and set \ifpst@rtedL to FALSE.

```
839 \ifpst@rtedL\else%
840 \list@clear{\inserts@list}%
841 \global\let\next@insert=\empty%
842 \global\pst@rtedLtrue%
843 \fi%
844 \begingroup\normal@pars%
```

When parallel processing we check that we haven't exceeded the maximum number of chunks. In any event we grab a box for the forthcoming text.

```
845 \global\advance\l@dnumpststartsL \one%
846 \ifnum\l@dnumpststartsL>\l@dc@maxchunks%
847 \led@err@TooManyPstarts%
848 \global\l@dnumpststartsL=\l@dc@maxchunks%
849 \fi%
850 \global\setnamebox{\l@dc@l@rawbox\the\l@dnumpststartsL}=\vbox\bgroup%
851 \ifautopar\else%
852 \ifnumberpstart%
853 \ifsidepstartnum%
854 \else%
855 \thepstartL%
856 \fi%
857 \fi%
858 \fi%
859 \hsize=\Lcolwidth%
860 \numberedpar@true%
861 \iflabeledpstart\protected@edef\@currentlabel%
862 {\p@pstartL\thepstartL}\fi%
```

Dump the optional arguments

```
863 \ifstrempy{#1}%
864 {\csgdef{before@pstartL@the\l@dnumpststartsL}{\at@every@pstart}}%
865 {\csgdef{before@pstartL@the\l@dnumpststartsL}{\noindent#1}}%
866 }

867 \newcommandx*{\pstartR}[1][1]{%
868 \if@nbreak%
869 \let\@oldnbreak\@nbreaktrue%
870 \else%
871 \let\@oldnbreak\@nbreakfalse%
872 \fi%
873 \@nbreaktrue%
874 \ifluatex%
875 \xdef\l@luatextextdir@R{\the\luatextextdir}%
876 \xdef\l@luatexpardir@R{\the\luatexpardir}%
877 \xdef\l@luatexbodydir@R{\the\luatexbodydir}%
878 \fi%
```

```

879 \ifnumberingR \else%
880   \led@err@PstartNotNumbered%
881   \beginnumberingR%
882 \fi%
883 \ifnumberedpar@%
884   \led@err@PstartInPstart%
885   \pendR%
886 \fi%
887 \ifpst@rtedR\else%
888   \list@clear{\inserts@listR}%
889   \global\let\next@insertR=\empty%
890   \global\pst@rtedRtrue%
891 \fi%
892 \begingroup\normal@pars%
893 \global\advance\l@dnumpstartsR \@ne%
894 \ifnum\l@dnumpstartsR>\l@dc@maxchunks%
895   \led@err@TooManyPstarts%
896   \global\l@dnumpstartsR=\l@dc@maxchunks%
897 \fi%
898 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
899   \ifautopar\else%
900     \ifnumberpstart%
901       \ifsidepstartnum\else%
902         \thepstartR%
903       \fi%
904     \fi%
905   \fi%
906 \hsize=\Rcolwidth%
907 \numberedpar@true%
908 \iflabelpstart\protected@edef\@currentlabel%
909   {\p@pstartR\thepstartR}\fi%
910 \ifstrempy{#1}%
911   {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
912   {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
913 }

```

`\pendL` `\pend` must be used to end a numbered paragraph. Again we need a version that knows about left parallel texts.

```

914 \newcommand*{\pendL}[1][1]{%
915   \ifnumbering \else%
916     \led@err@PendNotNumbered%
917   \fi%
918   \ifnumberedpar@ \else%
919     \led@err@PendNoPstart%
920   \fi%

```

We set all the usual interline penalties to zero and then immediately call `\endgraf` to end the paragraph; this ensures that there'll be no large interline penalties to prevent us from slicing the paragraph into pieces. These penalties revert to the values that you set when the group for the `\vbox` ends.

```

921 \l@dzeropenalties%
922 \endgraf\global\num@lines=\prevgraf\egroup%
923 \global\par@line=0%

  End the group that was begun in the \pstart.
924 \endgroup%
925 \ignorespaces%
926 \@oldnbreak%
927 \dump@pstartL@pc%
928 \ifnumberpstart%
929   \addtocounter{pstartL}{1}%
930 \fi
931 \parledgroup@beforenotes@save{L}%

  Dump content of the optional argument.
932 \ifstrempy{#1}%
933   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
934   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
935 }

```

`\pendR` The version of `\pend` needed for right texts.

```

936 \newcommandx*{\pendR}[1][1]{%
937   \ifnumberingR \else%
938     \led@err@PendNotNumbered%
939   \fi%
940   \ifnumberedpar@ \else%
941     \led@err@PendNoPstart%
942   \fi%
943   \l@dzeropenalties%
944   \endgraf\global\num@linesR=\prevgraf\egroup%
945   \global\par@lineR=0%
946   \endgroup%
947   \ignorespaces%
948   \@oldnbreak%
949   \dump@pstartR@pc%
950   \ifnumberpstart%
951     \addtocounter{pstartR}{1}%
952   \fi%
953   \parledgroup@beforenotes@save{R}%
954   \ifstrempy{#1}%
955     {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
956     {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
957 }
958

```

`\ifprint@last@after@pendL` Two booleans set to true, when the time is to print the last optional argument of
`\ifprint@last@after@pendR` a `\pend`.

```

959 \newif\ifprint@last@after@pendL%
960 \newif\ifprint@last@after@pendR%

```

18.2 Processing one line

For parallel texts we have to be able to process left and right lines independently. For sequential text we happily use the original `\do@line`. Otherwise ...

```

\l@dleftbox  A line of left text will be put in the box \l@dleftbox, and analogously for a line
\l@drightbox of right text.
961 \newbox\l@dleftbox
962 \newbox\l@drightbox
963

\countLline  We need to know the number of lines processed.
\countRline 964 \newcount\countLline
965   \countLline \z@
966 \newcount\countRline
967   \countRline \z@
968

\@donereallinesL  We need to know the number of ‘real’ lines output (i.e., those that have been input
\@donetotallinesL by the user), and the total lines output (which includes any blank lines output for
\@donereallinesR synchronisation).
\@donetotallinesR 969 \newcount\@donereallinesL
970 \newcount\@donetotallinesL
971 \newcount\@donereallinesR
972 \newcount\@donetotallinesR
973

\do@lineL  The \do@lineL macro is called to do all the processing for a single line of left text.

974 \newcommand*{\do@lineL}{%
975   \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
976   \advance\countLline \@ne
977   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
978   {\vbadness=10000
979    \splittopskip=\z@
980    \do@lineLhook
981    \l@emptyd@ta
982    \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}
983                                     to\baselineskip}%
984   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startL}{\}
985   \unvbox\one@line \global\setbox\one@line=\lastbox
986   \getline@numL
987   \ifnum\@lock>\@ne%
988     \inserthangingsymboltrue%
989   \else%
990     \inserthangingsymbolfalse%
991   \fi
992   \setbox\l@dleftbox

```

```

993 \hb@xt@ \Lcolwidth{%
994   \affixline@num
995   \xifinlist{\the\l@dpscL}{\eled@sections@@}%
996   {\add@inserts\affixside@note}%
997   {\print@lineL}}%
998 \add@penaltiesL
999 \global\advance\@donereallinesL\@ne
1000 \global\advance\@donetotallinesL\@ne
1001 \else
1002   \setbox\l@leftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
1003   \global\advance\@donetotallinesL\@ne
1004 \fi}
1005
1006

```

`\print@lineL` `\print@lineL` is for lines without a sectioning command. See `eledmac` definition of `\print@line` for handbook.

```

1007 \def\print@lineL{%
1008   \affixpstart@numL%
1009   \l@dld@ta %space kept for backward compatibility
1010   \add@inserts\affixside@note%
1011   \l@dlsn@te %space kept for backward compatibility
1012   {\ledllfill\hb@xt@ \wd\one@line{%
1013     \do@insidelineLhook%
1014     \ifluatex%
1015       \luatextextdir\l@luatextextdir@L%
1016     \fi%
1017     \new@lineL%
1018     \inserthangingsymbolL%
1019     \l@dunhbox@line{\one@line}}\correcthangingL\ledrlfill\l@drd@ta%
1020   \l@drsn@te}}
1021

```

`\print@eledsectionL` `\print@eledsectionL` is for line with macro code.

```

1022 \def\print@eledsectionL{%%
1023   \addtocounter{pstartL}{-1}%
1024   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
1025   \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}%
1026   \numdef{\temp@}{\l@dpscL-1}%
1027   \xifinlist{\temp@}{\eled@sections@@}{\@nobreaktrue}{\@nobreakfalse}%
1028   \@eled@sectioningtrue%
1029   \bgroup%
1030     \ifluatex%
1031       \luatextextdir\l@luatextextdir@L%
1032       \luatexpardir\l@luatexpardir@L%
1033       \luatexbodydir\l@luatexbodydir@L%
1034     \fi%
1035     \csuse{eled@sectioning@\the\l@dpscL}%
1036   \egroup%
1037   \@eled@sectioningfalse%

```

```

1038 \global\csundef{eled@sectioning@the\l@dpscl}%
1039 \if@RTL%
1040 \hspace{-3\paperwidth}%
1041 {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1042 \else%
1043 \hspace{3\paperwidth}%
1044 {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1045 \fi%
1046 \vskip\eledsection@correcting@skip%
1047 }
1048

```

`\dolineLhook` These high-level commands just redefine the low-level commands. They have to be used by user, without `\makeatletter`.

```

\dolineRhook 1049 \newcommand*{\dolineLhook}[1]{\gdef\dolineLhook{#1}}%
\dolineRhook 1050 \newcommand*{\dolineRhook}[1]{\gdef\dolineRhook{#1}}%
1051 \newcommand*{\doinsidelineLhook}[1]{\gdef\doinsidelineLhook{#1}}%
1052 \newcommand*{\doinsidelineRhook}[1]{\gdef\doinsidelineRhook{#1}}%
1053

```

`\do@lineLhook` Hooks, initially empty, into the respective `\do@line(L/R)` macros.

```

\dolineRhook 1054 \newcommand*{\do@lineLhook}{%}
\dolineRhook 1055 \newcommand*{\do@lineRhook}{%}
\dolineRhook 1056 \newcommand*{\do@insidelineLhook}{%}
\dolineRhook 1057 \newcommand*{\do@insidelineRhook}{%}
1058

```

`\do@lineR` The `\do@lineR` macro is called to do all the processing for a single line of right text.

```

1059 \newcommand*{\do@lineR}{%
1060 \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscl}%
1061 \ledRcol@true%
1062 \advance\countRline \@ne
1063 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscl}%
1064 {\vbadness=10000
1065 \splittopskip=\z@
1066 \do@lineRhook
1067 \l@emptyd@ta
1068 \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscl}
1069 to\baselineskip}%
1070 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startR}{%}
1071 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
1072 \getline@numR
1073 \ifnum\@lockR>\@ne%
1074 \inserthangingsymbolRtrue
1075 \else%
1076 \inserthangingsymbolRfalse%
1077 \fi%

```



```

1078 \setbox\l@drightbox
1079 \hb@xt@ \Rcolwidth{%
1080   \affixline@numR%
1081   \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1082   {\add@insertsR\affixside@noteR}%
1083   {\print@lineR}%
1084 }%
1085 \add@penaltiesR
1086 \global\advance\@donereallinesR\@ne
1087 \global\advance\@donetotallinesR\@ne
1088 \else
1089 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}
1090 \global\advance\@donetotallinesR\@ne
1091 \fi
1092 \ledRcol@false%
1093 }
1094
1095

```

```

\print@lineR
\print@eledsectionR

```

18.3 Line and page number computation

`\getline@numR` The `\getline@numR` macro determines the page and line numbers for the right text line we're about to send to the vertical list.

```

1096 \newcommand*{\getline@numR}{%
1097   \global\advance\absline@numR \@ne
1098   \do@actionsR
1099   \do@ballastR
1100   \ifledgroupnotesR\else\ifnumberline
1101     \ifsublines@
1102       \ifnum\sub@lockR<\tw@
1103         \global\advance\subline@numR \@ne
1104       \fi
1105     \else
1106       \ifnum\@lockR<\tw@
1107         \global\advance\line@numR \@ne
1108         \global\subline@numR \z@
1109       \fi
1110     \fi
1111   \fi
1112 \fi
1113 }
1114 \newcommand*{\getline@numL}{%
1115   \global\advance\absline@num \@ne
1116   \do@actions
1117   \do@ballast
1118   \ifledgroupnotesL\else\ifnumberline
1119     \ifsublines@

```

```

1120     \ifnum\sub@lock<\tw@
1121         \global\advance\subline@num \@ne
1122     \fi
1123 \else
1124     \ifnum\@lock<\tw@
1125         \global\advance\line@num \@ne
1126         \global\subline@num \z@
1127     \fi
1128 \fi
1129 \fi
1130 \fi
1131 }
1132
1133

```

`\do@ballastR` The real work in the line macros above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballastR` out of the way.

```

1134 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1135 \begingroup
1136     \advance\absline@numR \@ne
1137     \ifnum\next@actionlineR=\absline@numR
1138         \ifnum\next@actionR>-1001
1139             \global\advance\ballast@count by -\c@ballast
1140         \fi
1141     \fi
1142 \endgroup}

```

`\do@actionsR` The `\do@actionsR` macro looks at the list of actions to take at particular right text absolute line numbers, and does everything that's specified for the current line.
`\do@actions@fixedcodeR`
`\do@actions@nextR`

It may call itself recursively and we use tail recursion, via `\do@actions@nextR` for this.

```

1143 \newcommand*{\do@actions@fixedcodeR}{%
1144     \ifcase\@l@tempcnta%
1145         \or% % 1001
1146             \global\sublines@true
1147         \or% % 1002
1148             \global\sublines@false
1149         \or% % 1003
1150             \global\@lockR=\@ne
1151         \or% % 1004
1152             \ifnum\@lockR=\tw@
1153                 \global\@lockR=\thr@@
1154             \else
1155                 \global\@lockR=\z@
1156             \fi
1157         \or% % 1005
1158             \global\sub@lockR=\@ne
1159         \or% % 1006

```

```

1160 \ifnum\sub@lockR=\tw@
1161 \global\sub@lockR=\thr@@
1162 \else
1163 \global\sub@lockR=\z@
1164 \fi
1165 \or% % 1007
1166 \l@dskipnumbertrue
1167 \else
1168 \led@warn@BadAction
1169 \fi}
1170
1171
1172 \newcommand*{\do@actionsR}{%
1173 \global\let\do@actions@nextR=\relax
1174 \@l@dttempcntb=\absline@numR
1175 \ifnum\@l@dttempcntb<\next@actionlineR\else
1176 \ifnum\next@actionR>-1001\relax
1177 \global\page@numR=\next@actionR
1178 \ifbypage@R
1179 \global\line@numR \z@ \global\subline@numR \z@
1180 \fi
1181 \else
1182 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1183 \@l@dttempcnta=-\next@actionR
1184 \advance\@l@dttempcnta by -5001\relax
1185 \ifsublines@
1186 \global\subline@numR=\@l@dttempcnta
1187 \else
1188 \global\line@numR=\@l@dttempcnta
1189 \fi
1190 \else
1191 \@l@dttempcnta=-\next@actionR
1192 \advance\@l@dttempcnta by -1000\relax
1193 \do@actions@fixedcodeR
1194 \fi
1195 \fi
1196 \ifx\actionlines@listR\empty
1197 \gdef\next@actionlineR{1000000}%
1198 \else
1199 \gl@p\actionlines@listR\to\next@actionlineR
1200 \gl@p\actions@listR\to\next@actionR
1201 \global\let\do@actions@nextR=\do@actionsR
1202 \fi
1203 \fi
1204 \do@actions@nextR}
1205

```

18.4 Line number printing

```

\l@dcalcnun \affixline@numR is the right text version of the \affixline@num macro.
\ch@cksub@l@ckR 1206
\ch@ck@l@ckR 1207 \providecommand*\l@dcalcnun}[3]{%
\fx@l@cksR 1208 \ifnum #1 > #2\relax
\affixline@numR 1209 \@l@tempcnta = #1\relax
1210 \advance\@l@tempcnta by -#2\relax
1211 \divide\@l@tempcnta by #3\relax
1212 \multiply\@l@tempcnta by #3\relax
1213 \advance\@l@tempcnta by #2\relax
1214 \else
1215 \@l@tempcnta=#2\relax
1216 \fi}
1217
1218 \newcommand*\ch@cksub@l@ckR}{%
1219 \ifcase\sub@lockR
1220 \or
1221 \ifnum\sublock@disp=\@ne
1222 \@l@tempcntb \z@ \@l@tempcnta \@ne
1223 \fi
1224 \or
1225 \ifnum\sublock@disp=\tw@
1226 \else
1227 \@l@tempcntb \z@ \@l@tempcnta \@ne
1228 \fi
1229 \or
1230 \ifnum\sublock@disp=\z@
1231 \@l@tempcntb \z@ \@l@tempcnta \@ne
1232 \fi
1233 \fi}
1234
1235 \newcommand*\ch@ck@l@ckR}{%
1236 \ifcase\@lockR
1237 \or
1238 \ifnum\lock@disp=\@ne
1239 \@l@tempcntb \z@ \@l@tempcnta \@ne
1240 \fi
1241 \or
1242 \ifnum\lock@disp=\tw@
1243 \else
1244 \@l@tempcntb \z@ \@l@tempcnta \@ne
1245 \fi
1246 \or
1247 \ifnum\lock@disp=\z@
1248 \@l@tempcntb \z@ \@l@tempcnta \@ne
1249 \fi
1250 \fi}
1251
1252 \newcommand*\fx@l@cksR}{%

```

```

1253 \ifcase\@lockR
1254 \or
1255   \global\@lockR \tw@
1256 \or \or
1257   \global\@lockR \z@
1258 \fi
1259 \ifcase\sub@lockR
1260 \or
1261   \global\sub@lockR \tw@
1262 \or \or
1263   \global\sub@lockR \z@
1264 \fi}
1265
1266
1267 \newcommand*{\affixline@numR}{%
1268 \ifledgroupnotesR\else\ifnumberline
1269 \ifl@dskipnumber
1270   \global\l@dskipnumberfalse
1271 \else
1272   \ifsublines@
1273     \l@dttempcntb=\subline@numR
1274     \l@dcalcnun{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1275     \ch@cksub@lockR
1276   \else
1277     \l@dttempcntb=\line@numR
1278     \ifx\linenumberlist\empty
1279       \l@dcalcnun{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1280     \else
1281       \l@dttempcnta=\line@numR
1282       \edef\rem@inder{\linenumberlist,\number\line@numR,}%
1283       \edef\sc@n@list{\def\noexpand\sc@n@list
1284         ###1,\number\l@dttempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1285       \sc@n@list\expandafter\sc@n@list\rem@inder|
1286       \ifx\rem@inder\empty\advance\l@dttempcnta\@ne\fi
1287     \fi
1288     \ch@ck@l@ckR
1289   \fi
1290 \ifnum\l@dttempcnta=\l@dttempcntb
1291   \if@twocolumn
1292     \if@firstcolumn
1293       \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1294     \else
1295       \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1296     \fi
1297   \else
1298     \l@dttempcntb=\line@marginR
1299     \ifnum\l@dttempcntb>\@ne
1300       \advance\l@dttempcntb by\page@numR
1301     \fi
1302     \ifodd\l@dttempcntb

```

```

1303      \gdef\l@drd@ta{\rlap{{\rightlinenumR}}}%
1304      \else
1305      \gdef\l@dld@ta{\llap{{\leftlinenumR}}}%
1306      \fi
1307  \fi
1308  \fi
1309  \f@x@l@cksR
1310 \fi
1311 \fi
1312 \fi}

```

18.5 Pstart number printing in side

The printing of the pstart number is like in eldmac, with two differences :

- Some commands have versions suffixed by R or L.
- The `\affixpstart@num` and `\affixpstart@numR` commands are called in the `\Pages` command. Consequently, the `pstartL` and `pstartR` counters must be reset at the beginning of this command.

```

\affixpstart@numL
\affixpstart@numR 1313
  \leftpstartnumR 1314 \newcommand*{\affixpstart@numL}{%
\rightpstartnumR 1315 \ifsidepstartnum
  \leftpstartnumL 1316 \if@twocolumn
\rightpstartnumL 1317   \if@firstcolumn
  \ifpstartnumR 1318     \gdef\l@dld@ta{\llap{{\leftpstartnumL}}}%
1319     \else
1320     \gdef\l@drd@ta{\rlap{{\rightpstartnumL}}}%
1321     \fi
1322   \else
1323     \l@dttempcntb=\line@margin
1324     \ifnum\l@dttempcntb>\@ne
1325       \advance\l@dttempcntb \page@num
1326     \fi
1327     \ifodd\l@dttempcntb
1328       \gdef\l@drd@ta{\rlap{{\rightpstartnumL}}}%
1329     \else
1330       \gdef\l@dld@ta{\llap{{\leftpstartnumL}}}%
1331     \fi
1332   \fi
1333 \fi
1334 }
1335 \newcommand*{\affixpstart@numR}{%
1336 \ifsidepstartnum
1337 \if@twocolumn
1338   \if@firstcolumn
1339     \gdef\l@dld@ta{\llap{{\leftpstartnumR}}}%
1340     \else

```

```

1341 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1342 \fi
1343 \else
1344 \l@dttempcntb=\line@marginR
1345 \ifnum\l@dttempcntb>\@ne
1346 \advance\l@dttempcntb \page@numR
1347 \fi
1348 \ifodd\l@dttempcntb
1349 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1350 \else
1351 \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1352 \fi
1353 \fi
1354 \fi
1355 }
1356
1357 \newcommand*{\leftpstartnumL}{
1358 \ifpstartnum
1359 \thepstartL
1360 \kern\linenumsep\global\pstartnumfalse\fi
1361 }
1362 \newcommand*{\rightpstartnumL}{
1363 \ifpstartnum\kern\linenumsep
1364 \thepstartL
1365 \global\pstartnumfalse\fi
1366 }
1367 \newif\ifpstartnumR
1368 \pstartnumRtrue
1369 \newcommand*{\leftpstartnumR}{
1370 \ifpstartnumR
1371 \thepstartR
1372 \kern\linenumsep\global\pstartnumRfalse\fi
1373 }
1374 \newcommand*{\rightpstartnumR}{
1375 \ifpstartnumR\kern\linenumsep
1376 \thepstartR
1377 \global\pstartnumRfalse\fi
1378 }

```

18.6 Add insertions to the vertical list

`\inserts@listR` `\inserts@listR` is the list macro that contains the inserts that we save up for one right text paragraph.

```
1379 \list@create{\inserts@listR}
```

`\add@insertsR` The right text version.

```

\add@inserts@nextR 1380 \newcommand*{\add@insertsR}{%
1381 \global\let\add@inserts@nextR=\relax
1382 \ifx\inserts@listR\empty \else

```

```

1383 \ifx\next@insertR\empty
1384 \ifx\insertlines@listR\empty
1385 \global\noteschanged@true
1386 \gdef\next@insertR{100000}%
1387 \else
1388 \gl@p\insertlines@listR\to\next@insertR
1389 \fi
1390 \fi
1391 \ifnum\next@insertR=\absline@numR
1392 \gl@p\inserts@listR\to\@insertR
1393 \@insertR
1394 \global\let\@insertR=\undefined
1395 \global\let\next@insertR=\empty
1396 \global\let\add@inserts@nextR=\add@insertsR
1397 \fi
1398 \fi
1399 \add@inserts@nextR}
1400

```

18.7 Penalties

`\add@penaltiesL` `\add@penaltiesL` is the last macro used by `\do@lineL`. It adds up the club, widow, and interline penalties, and puts a single penalty of the appropriate size back into the paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

In the code below, which is a virtual copy of the original `\add@penalties`, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we're working on at the moment. The count `\@l@dttempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast`. Finally, the penalty is checked to see that it doesn't go below -10000 .

```

\newcommand*{\add@penaltiesR}{\@l@dttempcnta=\ballast@count
\ifnum\num@linesR>\@ne
\global\advance\par@lineR \@ne
\ifnum\par@lineR=\@ne
\advance\@l@dttempcnta by \clubpenalty
\fi
\@l@dttempcntb=\par@lineR \advance\@l@dttempcntb \@ne
\ifnum\@l@dttempcntb=\num@linesR
\advance\@l@dttempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
\advance\@l@dttempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@dttempcnta=\z@
\relax

```



```

\else
  \ifnum\l@dttempcnta>-10000
    \penalty\l@dttempcnta
  \else
    \penalty -10000
  \fi
\fi}

```

This is for a single chunk. However, as we are probably dealing with several chunks at a time, the above is not really relevant. I think that it is likely with parallel text that there is no real need to add back any penalties; even if there was, they would have to match across the left and right lines. So, I end up with the following.

```

1401 \newcommand*\add@penaltiesL-{}
1402 \newcommand*\add@penaltiesR-{}
1403

```

18.8 Printing leftover notes

`\flush@notesR` The `\flush@notesR` macro is called after the entire right text has been sliced up and sent on to the vertical list.

```

1404 \newcommand*\flush@notesR-{}
1405   \xloop
1406     \ifx\inserts@listR\empty \else
1407       \gl@p\inserts@listR\to\@insertR
1408       \@insertR
1409       \global\let\@insertR=\undefined
1410     \repeat}
1411

```

19 Footnotes

19.1 Normal footnote formatting

The `\printlines` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on page ??: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

`\printlinesR` This is the right text version of `\printlines` and takes account of `\Rlineflag`.
`\ledsavedprintlines` Just in case, `\ledsavedprintlines` is a copy of the original `\printlines`.
 Just a reminder of the arguments:

<code>\printlinesR</code>	<code>#1</code>	<code> </code>	<code>#2</code>	<code> </code>	<code>#3</code>	<code> </code>	<code>#4</code>	<code> </code>	<code>#5</code>	<code> </code>	<code>#6</code>	<code> </code>	<code>#7</code>
<code>\printlinesR</code>	start-page		line		subline		end-page		line		subline		font

```

1412 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1413   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%

```

```

1414 \ifl@d@pnum #1\fullstop\fi
1415 \ifledplinenum \linenumr@p{#2}\Rlineflag\else \sympmlinenum\fi
1416 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1417 \ifl@d@dash \endashchar\fi
1418 \ifl@d@pnum #4\fullstop\fi
1419 \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1420 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1421 \endgroup}
1422
1423 \let\ledsavedprintlines\printlines
1424

```

19.2 Footnotes output specific to \Pages

`\print@Xnotes@forpages` The `\onlyXside` and `\onlysideX` hooks for `\Pages` allow notes to be printed either in left or right pages only. The implementation of such features is delegated to `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed²:

- If notes are to be printed in both sides, we just proceed the usual way: print the foot starts for the series, then the foot group.
- If notes are to be printed in the left side, we do these prints only for even pages ; if notes are to be printed in the right side, we do these prints only for odd pages.
- However, that is not enough. Because the problem does not only consists in printing notes in any particular page. It is also not to put aside room for notes in the pages where we don't want to print them. To take an example: if some note in the left side is too long by 160pt to be printed in full in the left page, we do not want to put aside 160pt a space for it in the following right page.
- To solve this problem, we change the magnification factor associated with notes before going to the next page. If we start a page where no notes are supposed to be printed, the magnification counter is set to 0. We also set the note skip to 0pt. Before starting a new page where these notes are supposed to be printed, we reset these counter and skip to their default values. (About these counter and skip, read *TeXbook* p. 122-125).
- There still remains a last problem. This problem is quite complex to understand, so an example will speak for itself. Suppose we allow 10 lines of notes by page. Suppose a long note, be it 25 lines, which needs three pages to be printed. Suppose it must be printed only on left pages, namely odd pages.

On p. 2, the first 10 lines of the notes are printed. On p. 3, the box associated to the notes contains 10 lines. However, as we are in a right page, we don't void this box. So \TeX will keep its content for the pages to come. However,

²See <http://tex.stackexchange.com/a/230332/7712>.

on p. 4 it will also add one line in the footnote box, because in any case, `TeX` adds some content in the box when preparing the output routines, even if there is some content left in this box from the previous pages. So the lines in the note box at p. 4 will be $10 + 1 = 11$. There is one line which should not be there. Furthermore, as the box size is for 10 lines and not for 11 lines, this last line will be glued to the previous one.

To fix this double issue:

- For the pages where notes must be NOT printed, we allow to every note box one line less than it ought to be. In our example, that means that we allow `TeX` to add only $10 - 1 = 9$ line in the note box on p. 3. Before shifting to the pages where notes must be printed, we allow to every notes the expected number of lines. In our example, that means that we allow `TeX` to add 10 lines in the note box on p. 4. As on p. 3 only 9 lines were allowed, that means note box of p. 4 will contain $9 + 1 = 10$ lines. So the “one line too many” problem is solved.
- Still remains the “glue” problem. We solve it by recreating a clean note box. We split the one which is created by `TeX` to get the next line printed. Then, we create the new box, by bringing together the first part and the last part of the splitted box, adding some skip between them. That is achieved by `\correct@Xfootins@box` (or `\correct@footinsX@box` for familiar notes).

The code to print critical notes, when processing `\Pages`

```
1425 \newcommand\print@Xnotes@forpages[1]{%
```

First case: notes are for both sides. Just print the note start and the note group

```
1426   \ifcseempty{onlyXside@#1}{}%
1427     \csuse{#1footstart}{#1}%
1428     \csuse{#1footgroup}{#1}%
1429   }%
```

Second case: notes are for one side only. First test if we are in a page where they must be printed.

```
1430   {%
1431     \ifboolexpr{%
1432       ((test {\ifcsstring{onlyXside@#1}{L}} and not test{\ifnumodd{\c@page}}))%
1433       or%
1434       (test {\ifcsstring{onlyXside@#1}{R}} and test{\ifnumodd{\c@page}}))%
1435     }%
```

If we are in a page where notes must be printed, print them, making box’s correction before.

```
1436   {%
1437     \correct@Xfootins@box{#1}%
1438     \csuse{#1footstart}{#1}%
1439     \csuse{#1footgroup}{#1}%
```

Then, say to not keep room for notes in the next page.

```
1440      \global\count\csuse{#1footins}=0%
1441      \global\skip\csuse{#1footins}=0pt%
```

And also, allow one line less for notes in the next page.

```
1442      \csuse{Xnotefontsize@#1}%
1443      \global\advance\dimen\csuse{#1footins} by -\baselineskip%
```

Now we have printed the notes. So we put aside this fact.

```
1444      \global\boolfalse{keepforXside@#1}%
1445      }%
```

In case we are on a page where notes must NOT be printed. First, memorize that we have not printed the notes, despite having some to print.

```
1446      {%
1447      \global\booltrue{keepforXside@#1}%
```

Then restore expected rooms for notes on the next page.

```
1448      \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1449      \global\skip\csuse{#1footins}=\csuse{beforeXnotes@#1}%
```

Last but not least, restore the normal line number allowed to notes for the following page.

```
1450      \bgroup%
1451      \csuse{Xnotefontsize@#1}%
1452      \global\advance\dimen\csuse{#1footins} by \baselineskip%
1453      \egroup%
1454 % End of \cs{print@Xnotes@forpages}.
1455      }%
1456      }%
1457 }%
```

Now, `\correct@Xfootins@box`, to fix problem of last line being glued to the previous one.

```
1458 \newcommand{\correct@Xfootins@box}[1]{%
```

We need to make correction only in case we have not printed any note in the previous page, although there was to be “normally” printed.

```
1459 \ifbool{keepforXside@#1}{%
```

Some setting needed to do the right splitting.

```
1460      \csuse{Xnotefontsize@#1}%
1461      \splittopskip=0pt%
```

And now, split the last line, and push in the right place.

```
1462      \global\setbox\csuse{#1footins}=\vbox{%
1463      \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1464      \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1465      \unvbox\csuse{#1footins}%
1466      }%
```

End of the macro.

```

1467 }{}%
1468 }%

And now, the same for familiar footnotes.

1469 \newcommand\print@notesX@forpages[1]{%
1470   \ifcseempty{onlysideX@#1}{%
1471     \csuse{footstart#1}{#1}%
1472     \csuse{footgroup#1}{#1}%
1473   }%
1474   {%
1475     \ifboolexpr{%
1476       ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page}}))%
1477       or%
1478       (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}}))%
1479     }%
1480     {%
1481       \correct@footinsX@box{#1}%
1482       \csuse{footstart#1}{#1}%
1483       \csuse{footgroup#1}{#1}%
1484       \global\count\csuse{footins#1}=0%
1485       \global\skip\csuse{footins#1}=0pt%
1486       \csuse{notefontsizeX@#1}%
1487       \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1488       \global\boolfalse{keepforsideX@#1}%
1489     }%
1490     {%
1491       \global\booltrue{keepforsideX@#1}%
1492       \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1493       \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1494       \bgroup%
1495         \csuse{notefontsizeX@#1}%
1496         \global\advance\dimen\csuse{footins#1} by \baselineskip%
1497       \egroup%
1498     }%
1499   }%
1500 }%

1501 \newcommand{\correct@footinsX@box}[1]{%
1502   \ifbool{keepforsideX@#1}{%
1503     \csuse{notefontsizeX@#1}%
1504     \splittopskip=0pt%
1505     \global\setbox\csuse{footins#1}=\vbox{%
1506       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1507       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1508       \unvbox\csuse{footins#1}%
1509     }%
1510   }{}%
1511 }%
```

20 Cross referencing

`\labelref@listR` Set up a new list, `\labelref@listR`, to hold the page, line and sub-line numbers for each label in right text.

```
1512 \list@create{\labelref@listR}
1513
```

`\edlabel` Since version 1.18.0, this command is defined only one time in `eledmac`, including features for `eledpar`.

`\l@dmake@labelsR` This is the right text version of `\l@dmake@labels`, taking account of `\Rlineflag`.

```
1514 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1515   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1516     \led@warn@DuplicateLabel{#4}%
1517   \fi
1518   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\Rlineflag|#3|#4}%
1519   \ignorespaces}
1520 \AtBeginDocument{%
1521   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%}
1522 }
1523
```

`\@lab` The `\@lab` command, which appears in the `\linenum@out` file, appends the current values of page, line and sub-line to the `\labelref@list`. These values are defined by the earlier `\@page`, `\@nl`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

```
1524 \renewcommand*{\@lab}{%
1525   \ifledRcol
1526     \xright@appenditem{\linenumr@p{\line@numR}}{|%
1527     \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1528     \to\labelref@listR
1529   \else
1530     \xright@appenditem{\linenumr@p{\line@num}}{|%
1531     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1532     \to\labelref@list
1533   \fi}
1534
```

21 Side notes

Regular `\marginpars` do not work inside numbered text — they don't produce any note but do put an extra unnumbered blank line into the text.

`\sidenote@marginR` Specifies which margin sidenotes can be in.

```
\sidenotemargin* 1535 \WithSuffix\newcommand\sidenotemargin*[1]{%
1536   \l@dmakesidenote@margin{#1}
1537   \global\sidenote@marginR=\@l@tempcntb
1538   \global\sidenote@margin=\@l@tempcntb
```

```

1539 }
1540 \newcount\sidenote@marginR
1541 \global\sidenote@margin=\@ne
1542

```

`\affixside@noteR` The right text version of `\affixside@note`.

```

1543 \newcommand*{\affixside@noteR}{%
1544   \def\sidenotecontent@{%
1545     \numdef{\itemcount@}{0}%
1546     \def\do##1{%
1547       \ifnumequal{\itemcount@}{0}%
1548         {%
1549           \appto\sidenotecontent@{##1}}% Not print not separator before the 1st note
1550         {\appto\sidenotecontent@{\sidenotesep ##1}%
1551         }%
1552       \numdef{\itemcount@}{\itemcount@+1}%
1553     }%
1554     \dolistloop{\l@dcstetext}%
1555     \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1556   \gdef\@templ@d{%
1557     \gdef\@templ@n{\l@dcstetext\l@dcstetext@l\l@dcstetext@r}%
1558     \ifx\@templ@d\@templ@n \else%
1559       \if@twocolumn%
1560         \if@firstcolumn%
1561           \setl@dlp@rbox{##1}{\sidenotecontent@}%
1562         \else%
1563           \setl@drp@rbox{\sidenotecontent@}%
1564         \fi%
1565       \else%
1566         \@l@tempcntb=\sidenote@marginR%
1567         \ifnum\@l@tempcntb>\@ne%
1568           \advance\@l@tempcntb by\page@numR%
1569         \fi%
1570         \ifodd\@l@tempcntb%
1571           \setl@drp@rbox{\sidenotecontent@}%
1572           \gdef\sidenotecontent@{%
1573             \numdef{\itemcount@}{0}%
1574             \dolistloop{\l@dcstetext@l}%
1575             \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1576             \setl@dlp@rbox{\sidenotecontent@}%
1577           \else%
1578             \setl@dlp@rbox{\sidenotecontent@}%
1579             \gdef\sidenotecontent@{%
1580               \numdef{\itemcount@}{0}%
1581               \dolistloop{\l@dcstetext@r}%
1582               \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1583               \setl@drp@rbox{\sidenotecontent@}%
1584             \fi%
1585           \fi%

```

```

1586 \fi%
1587 }
1588

```

22 Familiar footnotes

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original `\@footnotetext`.

```

1589 \renewcommand{\l@dbfnote}[1]{%
1590   \ifnumberedpar@
1591   \gdef\@tag{#1\relax}%
1592   \ifledRcol%
1593     \xright@appenditem{\noexpand\vl@dbfnote{\@expandonce\@tag}}{\@thefnmark}}%
1594     \to\inserts@listR
1595     \global\advance\insert@countR \@ne%
1596   \else%
1597     \xright@appenditem{\noexpand\vl@dbfnote{\@expandonce\@tag}}{\@thefnmark}}%
1598     \to\inserts@list
1599     \global\advance\insert@count \@ne%
1600   \fi
1601 \fi\ignorespaces}
1602

```

`\normalbfnoteX`

```

1603 \renewcommand{\normalbfnoteX}[2]{%
1604   \ifnumberedpar@
1605   \ifledRcol%
1606     \ifluatex
1607       \footnotelang@lua[R]%
1608     \fi
1609     \@ifundefined{xpg@main@language}%if polyglossia
1610     {}%
1611     {\footnotelang@poly[R]}%
1612     \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1613     \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@expandonce\thisfootnote}}%
1614     \to\inserts@listR
1615     \global\advance\insert@countR \@ne%
1616   \else%
1617     \ifluatex
1618       \footnotelang@lua%
1619     \fi
1620     \@ifundefined{xpg@main@language}%if polyglossia
1621     {}%
1622     {\footnotelang@poly}%
1623     \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1624     \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\@expandonce\thisfootnote}}%
1625     \to\inserts@list
1626     \global\advance\insert@count \@ne%

```



```

1627     \fi
1628     \fi\ignorespaces}
1629

```

23 Verse

Like in eledmac, the insertion of hangingsymbol is base on \ifinserthangingsymbol, and, for the right side, on \ifinserthangingsymbolR.

```

\inserthangingsymbolL
\inserthangingsymbolR 1630 \newif\ifinserthangingsymbolR
1631 \newcommand{\inserthangingsymbolL}{%
1632 \ifinserthangingsymbol%
1633     \ifinstanzaL%
1634         \hangingsymbol%
1635     \fi%
1636 \fi}
1637 \newcommand{\inserthangingsymbolR}{%
1638 \ifinserthangingsymbolR%
1639     \ifinstanzaR%
1640         \hangingsymbol%
1641     \fi%
1642 \fi}

```

When a verse is hanged, the column separator is shifted. To prevent it, the \do@lineL and \do@lineR commands call \correcthangingL and \correcthangingR commands. These commands insert horizontal skip which length is equal to the hang indent.

```

\correcthangingL
\correcthangingR 1643 \newcommand{\correcthangingL}{%
1644 \ifl@dpaging\else%
1645     \ifinstanzaL%
1646         \ifinserthangingsymbol%
1647             \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1648                 \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1649         \fi%
1650     \fi%
1651 \fi}
1652
1653 \newcommand{\correcthangingR}{%
1654 \ifl@dpaging\else%
1655     \ifinstanzaR%
1656         \ifinserthangingsymbolR%
1657             \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1658                 \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1659         \fi%
1660     \fi%
1661 \fi}

```

Before we can define the main stanza macros we need to be able to save and reset the category code for &. To save the current value we use `\next` from the `\loop` macro.

```
1662 \chardef\next=\catcode'\&
1663 \catcode'\&=\active
1664
```

astanza This is roughly an environmental form of `\stanza`, which treats its stanza-like contents as a single chunk.

```
1665 \newenvironment{astanza}{%
1666   \startstanzahook
1667   \catcode'\&\active
1668   \global\stanza@count\@ne\stanza@modulo\@ne
1669   \ifnum\usernamecount{sza@0@}=\z@
1670     \let\stanza@hang\relax
1671     \let\endlock\relax
1672   \else
1673     \rightskip\z@ plus 1fil\relax
1674   \fi
1675   \ifnum\usernamecount{szp@0@}=\z@
1676     \let\sza@penalty\relax
1677   \fi
1678   \def&{%
1679     \endlock\mbox{}}%
1680     \sza@penalty
1681     \global\advance\stanza@count\@ne
1682     \@astanza@line}%
1683   \def\&{\@stopastanza}%
1684   \pstart
1685   \@astanza@line
1686 }{}
1687
```

\@stopastanza This command is called by `\&` in `astanza` environment. It allows optional arguments.

```
1688 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1689   \endlock\mbox{}}%
1690   \pend[#1]%
1691   \endstanzaextra%
1692 }%
```

\@astanza@line This gets put at the start of each line in the environment. It sets up the paragraph style — each line is treated as a paragraph.

```
1693 \newcommand*{\@astanza@line}{%
1694   \ifnum\value{stanzaindentrepetition}=0
1695     \parindent=\csname sza@\number\stanza@count
1696       @\endcsname\stanzaindentbase
1697   \else
1698     \parindent=\csname sza@\number\stanza@modulo
```

```

1699             @\endcsname\stanzaindentbase
1700         \managestanza@modulo
1701     \fi
1702 \par
1703 \stanza@hang%\mbox{}%
1704 \ignorespaces}
1705

```

Lastly reset the modified category codes.

```

1706 \catcode'\&=\next
1707

```

24 Naming macros

The L^AT_EX kernel provides `\@namedef` and `\@namuse` for defining and using macros that may have non-letters in their names. We need something similar here as we are going to need and use some numbered boxes and counters.

`\newnamebox` A set of macros for creating and using ‘named’ boxes; the macros are called after the regular box macros, but including the string ‘name’.

```

\unhnamebox 1708 \providecommand*\newnamebox}[1]{%
\unvnamebox 1709 \expandafter\newbox\csname #1\endcsname}
\namebox 1710 \providecommand*\setnamebox}[1]{%
    1711 \expandafter\setbox\csname #1\endcsname}
    1712 \providecommand*\unhnamebox}[1]{%
    1713 \expandafter\unhbox\csname #1\endcsname}
    1714 \providecommand*\unvnamebox}[1]{%
    1715 \expandafter\unvbox\csname #1\endcsname}
    1716 \providecommand*\namebox}[1]{%
    1717 \csname #1\endcsname}
    1718

```

`\newnamecount` Macros for creating and using ‘named’ counts.

```

\usenamecount 1719 \providecommand*\newnamecount}[1]{%
    1720 \expandafter\newcount\csname #1\endcsname}
    1721 \providecommand*\usenamecount}[1]{%
    1722 \csname #1\endcsname}
    1723

```

25 Counts and boxes for parallel texts

In sequential text, each chunk (that enclosed by `\pstart ... \pend`) is put into a box called `\raw@text` and then immediately printed, resulting in the box being emptied and ready for the next chunk. For parallel processing multiple boxes are needed as printing is delayed. We also need extra counters for various things.

`\maxchunks` The maximum number of chunk pairs before printing has to be called for. The
`\l@dc@maxchunks` default is 5120 chunk pairs.

```

1724 \newcount\l@dc@maxchunks
1725 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1726   \maxchunks{5120}
1727

```

`\l@dnumpstartsL` The numbers of left and right chunks. `\l@dnumpstartsL` is defined in `eledmac`.
`\l@dnumpstartsR`

```

1728 \newcount\l@dnumpstartsR
1729

```

`\l@pscl` A couple of scratch counts for use in left and right texts, respectively.
`\l@pscr`

```

1730 \newcount\l@dpscL
1731 \newcount\l@dpscR
1732

```

`\l@dsetuprawboxes` This macro creates `\maxchunks` pairs of boxes for left and right chunks. The boxes
are called `\l@dLcolrawbox1`, `\l@dLcolrawbox2`, etc.

```

1733 \newcommand*{\l@dsetuprawboxes}{%
1734   \@l@tempcntb=\l@dc@maxchunks
1735   \loop\ifnum\@l@tempcntb>\z@
1736     \newnamebox{\l@dLcolrawbox\the\@l@tempcntb}
1737     \newnamebox{\l@dRcolrawbox\the\@l@tempcntb}
1738     \advance\@l@tempcntb \m@ne
1739   \repeat}
1740

```

`\l@dsetupmaxlinecounts` To be able to synchronise left and right texts we need to know the maximum num-
`\l@dzeromaxlinecounts` ber of text lines there are in each pair of chunks. `\l@dsetupmaxlinecounts` creates
`\maxchunks` new counts called `\l@dmaxlinesinpar1`, etc., and `\l@dzeromaxlinecounts`
zeroes all of them.

```

1741 \newcommand*{\l@dsetupmaxlinecounts}{%
1742   \@l@tempcntb=\l@dc@maxchunks
1743   \loop\ifnum\@l@tempcntb>\z@
1744     \newnamecount{\l@dmaxlinesinpar\the\@l@tempcntb}
1745     \advance\@l@tempcntb \m@ne
1746   \repeat}
1747 \newcommand*{\l@dzeromaxlinecounts}{%
1748   \begingroup
1749   \@l@tempcntb=\l@dc@maxchunks
1750   \loop\ifnum\@l@tempcntb>\z@
1751     \global\usenamecount{\l@dmaxlinesinpar\the\@l@tempcntb}=\z@
1752     \advance\@l@tempcntb \m@ne
1753   \repeat
1754   \endgroup}
1755

```

Make sure that all these are set up. This has to be done after the user has had an opportunity to change `\maxchunks`.

```

1756 \AtBeginDocument{%
1757   \l@dsetuprawboxes
1758   \l@dsetupmaxlinecounts
1759   \l@dzeromaxlinecounts
1760   \l@dnumpstartsL=\z@
1761   \l@dnumpstartsR=\z@
1762   \l@dpscL=\z@
1763   \l@dpscR=\z@}
1764

```

26 Fixing babel

With parallel texts there is the possibility that the two sides might use different languages via `babel`. On the other hand, `babel` might not be called at all (even though it might be already built into the format).

With the normal sequential text each line is initially typeset in the current language environment, and then it is output at which time its attachments are typeset (in the same language environment. In the parallel case lines are typeset in their current language but an attachment might be typeset outside the language environment of its line if the left and right side languages are different. To counter this, we have to make sure that the correct language is used at the proper times.

```

\ifl@dusedbabel A flag for checking if babel has been used as a package.
\l@dusedbabelfalse 1765 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1766 \l@dusedbabelfalse

\ifl@dsamelang Suppress \ifl@dsamelang which didn't work and was not logical, because both
columns could have the same language but not the main language of the document.

\l@dchecklang

\l@dbbl@set@language In babel the macro \bbl@set@language{<lang>} does the work when the language
<lang> is changed via \selectlanguage. Unfortunately for me, if it is given an
argument in the form of a control sequence it strips off the \ character rather than
expanding the command. I need a version that accepts an argument in the form
\lang without it stripping the \.

1767 \newcommand*{\l@dbbl@set@language}[1]{%
1768   \edef\languagename{#1}%
1769   \select@language{\languagename}%
1770   \if@files
1771     \protected@write\auxout{{\string\select@language{\languagename}}}%
1772     \addtocontents{toc}{{\string\select@language{\languagename}}}%
1773     \addtocontents{lof}{{\string\select@language{\languagename}}}%
1774     \addtocontents{lot}{{\string\select@language{\languagename}}}%
1775   \fi}
1776

```

The rest of the setup has to be postponed until the end of the preamble when we know if `babel` has been used or not. However, for now assume that it has not been used.

`\selectlanguage` `\selectlanguage` is a `babel` command. `\theledlanguageL` and `\theledlanguageR` are the names of the languages of the left and right texts. `\l@duselanguage` is similar to `\selectlanguage`.

```
\theledlanguageR 1777 \providecommand{\selectlanguage}[1]{%
1778 \newcommand*{\l@duselanguage}[1]{%
1779 \gdef\theledlanguageL{}
1780 \gdef\theledlanguageR{}
1781
```

Now do the `babel` fix or `polyglossia`, if necessary.

```
1782 \AtBeginDocument{%
1783   \@ifundefined{xp@main@language}{%
1784     \@ifundefined{bbl@main@language}{%
```

Either `babel` has not been used or it has been used with no specified language.

```
1785   \l@dusedbabelfalse
1786   \renewcommand*{\selectlanguage}[1]{}%
```

Here we deal with the case where `babel` has been used. `\selectlanguage` has to be redefined to use our version of `\bbl@set@language` and to store the left or right language.

```
1787   \l@dusedbabeltrue
1788   \let\l@doldselectlanguage\selectlanguage
1789   \let\l@doldbbl@set@language\bbl@set@language
1790   \let\bbl@set@language\l@dbbl@set@language
1791   \renewcommand{\selectlanguage}[1]{%
1792     \l@doldselectlanguage{#1}%
1793     \ifledRcol \gdef\theledlanguageR{#1}%
1794     \else      \gdef\theledlanguageL{#1}%
1795     \fi}
```

`\l@duselanguage` simply calls the original `\selectlanguage` so that `\theledlanguageL` and `\theledlanguageR` are unaltered.

```
1796   \renewcommand*{\l@duselanguage}[1]{%
1797     \l@doldselectlanguage{#1}}
```

Lastly, initialise the left and right languages to the current `babel` one.

```
1798   \gdef\theledlanguageL{\bbl@main@language}%
1799   \gdef\theledlanguageR{\bbl@main@language}%
1800   }%
1801 }
```

If on `Polyglossia`

```
1802 { \let\old@otherlanguage\otherlanguage%
1803   \renewcommand{\otherlanguage}[2][1]{%
1804     \selectlanguage{#1}{#2}%
1805     \ifledRcol \gdef\theledlanguageR{#2}%
```

```

1806         \else           \gdef\theledlanguageL{#2}%
1807         \fi}%
1808         \let\l@duselanguage\select@language%
1809         \gdef\theledlanguageL{\xpg@main@language}%
1810         \gdef\theledlanguageR{\xpg@main@language}%
    That's it.
1811 }}

```

```

    \if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
    \@pstartstrue 1812 \newif\if@pstarts
    \@pstartstrue 1813 \newcommand*{\check@pstarts}{%
    \check@pstarts 1814 \@pstartstrue
    1815 \ifnum\l@dnumpstartsL>\l@dpscL
    1816 \@pstartstrue
    1817 \else
    1818 \ifnum\l@dnumpstartsR>\l@dpscR
    1819 \@pstartstrue
    1820 \fi
    1821 \fi
    1822 }
    1823

```

```

    \ifaraw@text \checkraw@text checks whether the current Left or Right box is void or not. If
    \araw@texttrue one or other is not void it sets \araw@texttrue, otherwise both are void and it
    \araw@textfalse sets \araw@textfalse.
    \checkraw@text 1824 \newif\ifaraw@text
    1825 \araw@textfalse
    1826 \newcommand*{\checkraw@text}{%
    1827 \araw@textfalse
    1828 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}
    1829 \araw@texttrue
    1830 \else
    1831 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}
    1832 \araw@texttrue
    1833 \fi
    1834 \fi
    1835 }
    1836

```

```

\@writelinesinparL These write the number of text lines in a chunk to the section files, and then
\@writelinesinparR afterwards zero the counter.
1837 \newcommand*{\@writelinesinparL}{%
1838 \edef\next{%
1839 \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1840 \next
1841 \global\@donereallinesL \z@}
1842 \newcommand*{\@writelinesinparR}{%
1843 \edef\next{%
1844 \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%

```

```

1845 \next
1846 \global\@donereallinesR \z@
1847

```

27 Parallel columns

`\@eledsectionL` The parbox `\@eledsectionL` and `\@eledsectionR` will keep the sections' title.

```

\@eledsectionR 1848 \newsavebox{\@eledsectionL}%
1849 \newsavebox{\@eledsectionR}%

```

`\Columns` The `\Columns` command results in the previous Left and Right texts being typeset in matching columns. There should be equal numbers of chunks in the left and right texts.

```

1850 \newcommand*{\Columns}{%
1851   \l@dprintingcolumnstrue%
1852   \eledsection@correcting@skip=-\baselineskip% Correction for sections' titles
1853   \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1854     \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
1855   \fi

```

Start a group and zero counters, etc.

```

1856 \begingroup
1857   \l@dzeropenalties
1858   \endgraf\global\num@lines=\prevgraf
1859   \global\num@linesR=\prevgraf
1860   \global\par@line=\z@
1861   \global\par@lineR=\z@
1862   \global\l@dpscL=\z@
1863   \global\l@dpscR=\z@

```

Check if there are chunks to be processed, and process them two by two (left and right pairs).

```

1864   \check@pstarts
1865   \loop\if@pstarts
1866     \global\pstartnumtrue
1867     \global\pstartnumRtrue

```

Increment `\l@dpscL` and `\l@dpscR` which here count the numbers of left and right chunks. Also restore the value of the public pstart counters.

```

1868   \global\advance\l@dpscL \@ne
1869   \global\advance\l@dpscR \@ne
1870   \restore@pstartL@pc%
1871   \restore@pstartR@pc%

```

Check if there is text yet to be processed in at least one of the two current chunks, and also whether the left and right languages are the same

```

1872   \checkraw@text
1873 {   \loop\ifaraw@text

```


Grab the next pair of left and right text lines and output them, swapping languages if they differ, adding section title if needed.

```

1874      \l@duselanguage{\theledlanguageL}%
1875      \do@lineL
1876      \xifinlist{\the\l@dpscL}{\eled@sections@@}
1877      {%
1878      \ifdefstring{\@eledsectmark}{L}%
1879      {\csuse{eled@sectmark@the\l@dpscL}%
1880      }{}}%
1881      \global\csundef{eled@sectmark@the\l@dpscL}%
1882      \savebox{\@eledsectionL}{\parbox[t][t]{\Lcolwidth}{\vbox{\print@eledsectionL}}}%\vb
1883      }%
1884      {}%
1885      \l@duselanguage{\theledlanguageR}%
1886      \do@lineR
1887      \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1888      {%
1889      \ifdefstring{\@eledsectmark}{R}%
1890      {\csuse{eled@sectmark@the\l@dpscR R}%
1891      }{}}%
1892      \global\csundef{eled@sectmark@the\l@dpscR R}%
1893      \savebox{\@eledsectionR}{\parbox[t][t]{\Rcolwidth}{\vbox{\print@eledsectionR}}}%\vb
1894      }%
1895      \hb@xt@ \hsize{%
1896      \ifdefstring{\columns@position}{L}{\hfill }%
1897      \unhbox\l@leftbox%
1898      \ifhbox\@eledsectionL%
1899      \usebox{\@eledsectionL}%
1900      \fi%
1901      \print@columnseparator%
1902      \unhbox\l@rightbox%
1903      \ifhbox\@eledsectionR%
1904      \usebox{\@eledsectionR}%
1905      \fi%
1906      \ifdefstring{\columns@position}{R}{\hfill}%
1907      }%
1908      \checkraw@text
1909      \checkverseL
1910      \checkverseR
1911      \checkpb@columns
1912      \repeat}

```

Having completed a pair of chunks, write the number of lines in each chunk to the respective section files. Increment pstart counters and reset line numbering if it's by pstart.

```

1913      \@writelinesinparL
1914      \@writelinesinparR
1915      \check@pstarts
1916      \ifbypstart@%
1917      \write\linenum@out{\string\@set[1]}

```

```

1918         \resetprevline@
1919     \fi
1920     \ifbypstart@R
1921         \write\linenum@outR{\string\@set[1]}
1922         \resetprevline@
1923     \fi
1924 \repeat

```

Having output all chunks, make sure all notes have been output, then zero counts ready for the next set of texts. The boolean tests for stanza are switched to false.

```

1925     \flush@notes
1926     \flush@notesR
1927 \endgroup

1928 \global\l@dpscL=\z@
1929 \global\l@dpscR=\z@
1930 \global\l@dnumstartsL=\z@
1931 \global\l@dnumstartsR=\z@
1932 \l@dprintingcolumnsfalse%
1933 \ignorespaces
1934 \global\instanzaLfalse
1935 \global\instanzaRfalse}
1936

```

`\print@columnseparator` `\print@columnseparator` prints the column separator, with surrounding spaces (as the user has set them). We use the \TeX `\ifdim` instead of `etoolbox` to avoid having `\hfill` in a `{}`, which deletes some space (but not much).

```

1937 \def\print@columnseparator{%
1938     \ifdim\beforecolumnseparator<0pt%
1939         \hfill%
1940     \else%
1941         \hspace{\beforecolumnseparator}%
1942     \fi%
1943     \columnseparator%
1944     \ifdim\aftercolumnseparator<0pt%
1945         \hfill%
1946     \else%
1947         \hspace{\beforecolumnseparator}%
1948     \fi%
1949 }%
1950 %\end{macrocode}
1951 % \end{macro}
1952 % \begin{macro}{\checkpb@columns}
1953 % \cs{checkpb@columns} prevent or make pagebreaking in columns, depending of the use of \
1954 %     \begin{macrocode}
1955
1956 \newcommand{\checkpb@columns}{%
1957     \newif\if@pb
1958     \newif\if@nopb
1959     \IfStrEq{\led@pb@setting}{before}{

```

```

1960 \numdef{\next@absline}{\the\absline@num+1}%
1961 \numdef{\next@abslineR}{\the\absline@numR+1}%
1962 \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
1963 \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
1964 \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
1965 \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
1966 }{}
1967 \IfStrEq{\led@pb@setting}{after}{
1968 \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
1969 \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
1970 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
1971 \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
1972 }{}
1973 \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
1974 \if@pb\pagebreak[4]\fi
1975 }

```

`\columnseparator` The separator between line pairs in parallel columns is in the form of a vertical rule extending a little below the baseline and with a height slightly greater than the `\baselineskip`. The width of the rule is `\columnrulewidth` (initially 0pt so the rule is invisible).

```

1976 \newcommand*{\columnseparator}{%
1977 \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1978 \newdimen\columnrulewidth
1979 \columnrulewidth=\z@
1980

```

`\columnspan` The position of the `\Columns` in a page. Default value is R. Stored in `\columns@position`.

```

1981 \newcommand*{\columnspan}[1]{%
1982 \xdef\columns@position{#1}%
1983 }%
1984 \xdef\columns@position{R}%

```

`\beforecolumnseparator` `\beforecolumnseparator` and `\aftercolumnseparator` lengths are defined to `-1pt`. If user changes them to a positive length, the lengths are used to define blank spaces before / after the column separator, instead of `\hfill`.

```

1985 \newlength{\beforecolumnseparator}%
1986 \setlength{\beforecolumnseparator}{-2pt}%
1987
1988 \newlength{\aftercolumnseparator}%
1989 \setlength{\aftercolumnseparator}{-2pt}%
1990

```

`\setwidthliketwocolumns@L` The `\setwidth...` macros are called in `\beginnumbering` in a **non-parallel** typesetting context, to fix the width of the lines to be vertically aligned with parallel columns. They are also called at the beginning of a note's group, if some options are enabled. The `\setposition...` macros are called in `\beginnumbering` in a **non- parallel** typesetting context to fix the position of the lines. The

`\setwidthliketwocolumns@C`

`\setwidthliketwocolumns@R`

`\setpositionliketwocolumns@L`

`\setpositionliketwocolumns@C`

`\setpositionliketwocolumns@R`

`\setnoteposition...` macros are called in `\xxxfootstart` in a **non-parallel** typesetting context to fix the position of notes block.

```

1991 \newcommand{\setwidthliketwocolumns@L}{%
1992 % Temporary dimension, initially equal to the standard hsize, i.e. text width
1993 %   \begin{macrocode}
1994   \newdimen\temp%
1995   \temp=\hsize%
1996   Hsize : Left + Right width
1997   \hsize=\Lcolwidth%
1998   \advance\hsize\Rcolwidth%
1999   Now, calculating the remaining space
2000   \advance\temp-\hsize%
2001   And multiply the hsize by 2/3 of this space
2002   \multiply\temp by 2%
2003   \divide\temp by 3%
2004   \advance\hsize\temp%
2005 }%
2006
2007
2008 \newcommand{\setnotespositionliketwocolumns@L}{%
2009 \renewcommand{\ledrlfill}{\hfill}%
2010 }%
2011
2012 \newcommand{\setwidthliketwocolumns@C}{%
2013 % Temporary dimension, initially equal to the standard hsize, i.e. text width
2014 %   \begin{macrocode}
2015   \newdimen\temp%
2016   \temp=\hsize%
2017   Hsize : Left + Right width
2018   \hsize=\Lcolwidth%
2019   \advance\hsize\Rcolwidth%
2020   Now, calculating the remaining space
2021   \advance\temp-\hsize%
2022   And multiply the hsize by 1/2 of this space
2023   \divide\temp by 2%
2024   \advance\hsize\temp%
2025 }%
2026
2027 \newcommand{\setpositionliketwocolumns@C}{%
2028 \doinsidelinehook{\hfill}%
2029 \renewcommand{\ledrlfill}{\hfill}%
2030 }%

```

```

2030 \newcommand{\setnotespositionliketwocolumns@C}{%
2031   \newdimen\temp%
2032   \newdimen\tempa%
2033   \temp=\hsize%
2034   \tempa=\Lcolwidth%
2035   \advance\tempa\Rcolwidth%
2036   \advance\temp-\tempa%
2037   \divide\temp by 2%
2038   \leftskip=\temp%
2039   \rightskip=-\temp%
2040 }%
2041
2042 \newcommand{\setwidthliketwocolumns@R}{%
    Temporary dimension, initially equal to the standard hsize, i.e. text width
2043   \newdimen\temp%
2044   \temp=\hsize%
    Hsize : Left + Right width
2045   \hsize=\Lcolwidth%
2046   \advance\hsize\Rcolwidth%
    Now, calculating the remaining space
2047   \advance\temp-\hsize%
    And multiply the hsize by 2/3 of this space
2048   \multiply\temp by 2%
2049   \divide\temp by 3%
2050   \advance\hsize\temp%
2051 }%
2052
2053 \newcommand{\setpositionliketwocolumns@R}{%
2054   \doinsidelinehook{\hfill}%
2055 }%
2056
2057 \newcommand{\setnotespositionliketwocolumns@R}{%
2058   \newdimen\temp%
2059   \newdimen\tempa%
2060   \temp=\hsize%
2061   \tempa=\Lcolwidth%
2062   \advance\tempa\Rcolwidth%
2063   \advance\temp-\tempa%
2064   \divide\temp by 2%
2065   \leftskip=\temp%
2066   \rightskip=-\temp%
2067 }%
2068

```

28 Parallel pages

This is considerably more complicated than parallel columns.

`\numpagelinesL` Counts for the number of lines on a left or right page, and the smaller of the
`\numpagelinesR` number of lines on a pair of facing pages.

```
\l@dminpagelines 2069 \newcount\numpagelinesL
                  2070 \newcount\numpagelinesR
                  2071 \newcount\l@dminpagelines
                  2072
```

`\Pages` The `\Pages` command results in the previous Left and Right texts being typeset on matching facing pages. There should be equal numbers of chunks in the left and right texts.

```
2073 \newcommand*{\Pages}{%
2074   \l@dprintingpagetrue%
2075   \eledsection@correcting@skip=-2\baselineskip% line correcting for section titles.
2076   \parledgroup@notespacing@set@correction
2077   \typeout{}
2078   \typeout{***** PAGES *****}
2079   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
2080     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2081   \fi
```

As `\Pages` must be called outside of the pages environment, we have to redefine the `\Lcolwidth` and `\Rcolwidth` lengths, to prevent false overfull hboxes.

```
2082 \setlength{\Lcolwidth}{\textwidth}%
2083 \setlength{\Rcolwidth}{\textwidth}%
```

Get onto an empty even (left) page, then initialise counters, etc.

```
2084 \cleartol@devenpage
2085 \begingroup
2086   \l@dzeropenalties
2087   \endgraf\global\num@lines=\prevgraf
2088   \global\num@linesR=\prevgraf
2089   \global\par@line=\z@
2090   \global\par@lineR=\z@
2091   \global\l@dpscL=\z@
2092   \global\l@dpscR=\z@
2093   \writtenlinesLfalse
2094   \writtenlinesRfalse
```

The footnotes are printed in way which is different way from the one expected in `eledmac`, as we may want to have the notes printed in one side only.

```
2095 \let\print@Xnotes\print@Xnotes@forpages%
2096 \let\print@notesX\print@notesX@forpages%
```

Check if there are chunks to be processed.

```
2097 \check@pstarts
2098 \loop\if@pstarts
```

Loop over the number of chunks, incrementing the chunk counts (`\l@dpscL` and `\l@dpscR` are chunk (box) counts.)

```
2099 \global\advance\l@dpscL \@ne
2100 \global\advance\l@dpscR \@ne
```

Calculate the maximum number of real text lines in the chunk pair, storing the result in the relevant \l@dmaxlinesinpar.

```

2101      \getlinesfromparlistL
2102      \getlinesfromparlistR
2103      \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2104      {\usernamecount{\l@dmaxlinesinpar\the\l@dpscL}}%
2105      \check@pstarts
2106      \repeat

```

Zero the counts again, ready for the next bit.

```

2107      \global\l@dpscL=\z@
2108      \global\l@dpscR=\z@

```

Get the number of lines on the first pair of pages and store the minimum in \l@dminpagelines.

```

2109      \getlinesfrompagelistL
2110      \getlinesfrompagelistR
2111      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2112      {\l@dminpagelines}%

```

Now we start processing the left and right chunks (\l@dpscL and \l@dpscR count the left and right chunks), starting with the first pair.

```

2113      \check@pstarts
2114      \if@pstarts

```

Increment the chunk counts to get the first pair. Restore also the value of public pstart counters.

```

2115      \global\advance\l@dpscL \@ne
2116      \global\advance\l@dpscR \@ne
2117      \restore@pstartL@pc%
2118      \restore@pstartR@pc%

```

We haven't processed any lines from these chunks yet, so zero the respective line counts.

```

2119      \global\@donereallinesL=\z@
2120      \global\@donetotallinesL=\z@
2121      \global\@donereallinesR=\z@
2122      \global\@donetotallinesR=\z@

```

Start a loop over the boxes (chunks).

```

2123      \checkraw@text
2124 %      \begingroup
2125 {      \loop\ifaraw@text

```

See if there is more that can be done for the left page and set up the left language.

```

2126      \checkpageL
2127      \l@duselanguage{\theledlanguageL}%
2128 {      \loop\ifl@dsamepage%

```

Process the next (left) text line, adding it to the page. Eventually, adds the optional argument of pstart.

```

2129         \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%}
2130         \csuse{before@pstartL@the\l@dpscl}%
2131         \global\csundef{before@pstartL@the\l@dpscl}
2132         \do@lineL
2133         \xifinlist{\the\l@dpscl}{\eled@sections@}
2134         {\print@eledsectionL}%
2135         {}%
2136         \advance\numpagelinesL \@ne

```

When using shiftedpstarts option, a \l@dleftbox with a null height is not printed. That means we do not insert blank lines at the end of a left chunk lower than the corresponding right chunk. However, a \l@dleftbox with a null height will advance the \pagetotal in any case. Because if we do not do this, the \checkpageL could let \ifl@pagefull to false, and consequently a \@lopL equal to 1000 could be written in the numbered file, even if all the lines actually needed for the current page have been printed. \l@dleftbox

```

2137         \ifshiftedpstarts
2138             \ifdim\ht\l@dleftbox>Opt\hb@xt@%
2139                 \hsize{\ledstrutL\unhbox\l@dleftbox}%
2140             \else%
2141                 \dimen0=\pagetotal%
2142                 \advance\dimen0 by \baselineskip%
2143                 \global\pagetotal=\dimen0%
2144             \fi%
2145         \else%
2146             \parledgroup@correction@notespacing{L}
2147             \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2148         \fi

```

Perhaps we have to move to the next (left) box. Check if we have got all we can onto the page. If not, repeat for the next line. Check if we have to print the optional argument of the last pend. Check if the page is full. Check if the verse is split in two subsequent pages. Check there is any forced page breaks.

```

2149         \get@nextboxL%
2150         \ifprint@last@after@pendL%
2151             \csuse{after@pendL@the\l@dpscl}%
2152             \global\csundef{after@pendL@the\l@dpscl}%
2153         \fi%
2154         \checkpageL%
2155         \checkverseL
2156         \checkpbl
2157         \repeat

```

That (left) page has been filled. Output the number of real lines on the page — if the page break is because the page has been filled with lines, use the actual number, otherwise the page has been ended early in order to synchronise with the facing page so use an impossibly large number.


```

2158      \ifl@dpagfull
2159        \@writelinesonpageL{\the\numpagelinesL}%
2160      \else
2161        \@writelinesonpageL{1000}%
2162      \fi

```

Reset to zero the left-page line count, clear the page to get onto the facing (odd, right) page, and reinitialize the accumulated dimension of interline correction for notes in parallel ledgroup.

```

2163      \numpagelinesL \z@
2164      \parledgroup@correction@notespacing@init
2165      \clearl@dleftpage }%

```

Now do the same for the right text.

```

2166      \checkpageR%
2167      \l@duselanguage{\theledlanguageR}%
2168 {
2169      \loop\ifl@dsamepage%
2170        \initnumbering@sectcountR
2171        \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{%
2172          \csuse{before@pstartR@the\l@dpscR}%
2173          \global\csundef{before@pstartR@the\l@dpscR}
2174          \do@lineR
2175          \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
2176            {\print@eledsectionR}%
2177            {}%
2178          \advance\numpagelinesR \@ne
2179          \ifshiftedpstarts
2180            \ifdim\ht\l@drightbox>0pt\hb@xt@%
2181              \hsize{\ledstrutR\unhbox\l@drightbox}%
2182            \else%
2183              \dimen0=\pagetotal%
2184              \advance\dimen0 by \baselineskip%
2185              \global\pagetotal=\dimen0%
2186            \fi%
2187          \else%
2188            \parledgroup@correction@notespacing{R}
2189            \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2190          \fi
2191          \get@nextboxR%
2192          \ifprint@last@after@pendR%
2193            \csuse{after@pendR@the\l@dpscR}%
2194            \global\csundef{after@pendR@the\l@dpscR}%
2195          \fi%
2196          \checkpageR%
2197          \checkverseR
2198          \checkpbR
2199          \repeat
2200      \ifl@dpagfull
2201        \@writelinesonpageR{\the\numpagelinesR}%
2202      \else

```

```

2202         \@writelinesonpageR{1000}%
2203     \fi
2204     \numpagelinesR=\z@
2205     \parledgroup@correction@notespacing@init
    The page is full, so move onto the next (left, odd) page and repeat left text
    processing.
2206     \clearl@drightpage}
    More to do? If there is we have to get the number of lines for the next pair of
    pages before starting to output them.
2207     \checkraw@text
2208     \ifaraw@text
2209         \getlinesfrompagelistL
2210         \getlinesfrompagelistR
2211         \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2212                             {\l@dminpagelines}%
2213     \fi
2214     \repeat}
    We have now output the text from all the chunks.
2215     \fi
    Make sure that there are no inserts hanging around.
2216     \flush@notes
2217     \flush@notesR
2218     \endgroup
    Zero counts ready for the next set of left/right text chunks. The boolean tests for
    stanza are switched to false.
2219     \global\l@dpscL=\z@
2220     \global\l@dpscR=\z@
2221     \global\l@dnumpestartsL=\z@
2222     \global\l@dnumpestartsR=\z@
2223     \global\instanzaLfalse
2224     \global\instanzaRfalse
2225     \l@dprintingpagesfalse%
2226     \finish@Pages@notes%Needed to prevent final notes overlap line number
2227     \ignorespaces}
2228

```

\finish@Pages@notes This macro ensures that all long notes are printed at the end of `\Pages` typesetting, and that there is no more long notes for the next pages.

```

2229 \newcommand{\finish@Pages@notes}{%
2230   \def\do##1{%
2231     \ifvoid\csuse{##1footins}%
2232       \ifvoid\csuse{footins##1}\else%
2233         \newpage\null%
2234         \listbreak%
2235       \fi%
2236     \else%

```

```

2237      \newpage\null%
2238      \listbreak%
2239      \fi%
2240  }%
2241  \dolistloop{\@series}%
2242 }%

```

`\ledstrutL` Struts inserted into leftand right text lines.

```

\ledstrutR 2243 \newcommand*{\ledstrutL}{\strut}
2244 \newcommand*{\ledstrutR}{\strut}
2245

```

`\cleartoevenpage` `\cleartoevenpage`, which is defined in the memoir class, is like `\clear(double)page` except that we end up on an even page. `\cleartol@devenpage` is similar except that it first checks to see if it is already on an empty page.

```

2246 \providecommand{\cleartoevenpage}[1][\@empty]{%
2247   \clearpage
2248   \ifodd\c@page\hbox{ }#1\clearpage\fi}
2249 \newcommand*{\cleartol@devenpage}{%
2250   \ifdim\pagetotal<\topskip% on an empty page
2251   \else
2252     \clearpage
2253   \fi
2254   \ifodd\c@page\hbox{ }\clearpage\fi}

```

`\clearl@dleftpage` `\clearl@dleftpage` and `\clearl@drighthouse` get us onto an odd and even page, respectively, checking that we end up on the subsequent page. Both commands use `\newpage` and not `\clearpage`. Because `\clearpage` prints all footnotes before the next page, even if it has to add new empty pages, while `\newpage` does not. And as we want notes started in the left page continue in the right page and *vice-versa*, we must use `\newpage` and not `\clearpage`

```

2255 \newcommand*{\clearl@dleftpage}{%
2256   \ifdim\pagetotal=0pt\hbox{ }\fi%
2257   \newpage%
2258   \ifodd\c@page\else
2259     \led@err@LeftOnRightPage
2260     \hbox{ }%
2261     \cleardoublepage
2262   \fi}
2263
2264 \newcommand*{\clearl@drighthouse}{%
2265   \ifdim\pagetotal=0pt\hbox{ }\fi%
2266   \newpage%
2267   \ifodd\c@page
2268     \led@err@RightOnLeftPage
2269     \hbox{ }%
2270     \cleartoevenpage
2271   \fi}
2272

```

`\getlinesfromparlistL` `\getlinesfromparlistL` gets the next entry from the `\linesinpar@listL` and
`\@cs@linesinparL` puts it into `\@cs@linesinparL`; if the list is empty, it sets `\@cs@linesinparL` to
`\getlinesfromparlistR` 0. Similarly for `\getlinesfromparlistR`.

```
\@cs@linesinparR 2273 \newcommand*\getlinesfromparlistL{%
2274   \ifx\linesinpar@listL\empty
2275     \gdef\@cs@linesinparL{0}%
2276   \else
2277     \gl@p\linesinpar@listL\to\@cs@linesinparL
2278   \fi}
2279 \newcommand*\getlinesfromparlistR{%
2280   \ifx\linesinpar@listR\empty
2281     \gdef\@cs@linesinparR{0}%
2282   \else
2283     \gl@p\linesinpar@listR\to\@cs@linesinparR
2284   \fi}
2285
```

`\getlinesfrompagelistL` `\getlinesfrompagelistL` gets the next entry from the `\linesonpage@listL` and
`\@cs@linesonpageL` puts it into `\@cs@linesonpageL`; if the list is empty, it sets `\@cs@linesonpageL`
`\getlinesfrompagelistR` to 1000. Similarly for `\getlinesfrompagelistR`.

```
\@cs@linesonpageR 2286 \newcommand*\getlinesfrompagelistL{%
2287   \ifx\linesonpage@listL\empty
2288     \gdef\@cs@linesonpageL{1000}%
2289   \else
2290     \gl@p\linesonpage@listL\to\@cs@linesonpageL
2291   \fi}
2292 \newcommand*\getlinesfrompagelistR{%
2293   \ifx\linesonpage@listR\empty
2294     \gdef\@cs@linesonpageR{1000}%
2295   \else
2296     \gl@p\linesonpage@listR\to\@cs@linesonpageR
2297   \fi}
2298
```

`\@writelinesonpageL` These macros output the number of lines on a page to the section file in the form
`\@writelinesonpageR` of `\@lopL` or `\@lopR` macros.

```
2299 \newcommand*\@writelinesonpageL}[1]{%
2300   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2301   \next}
2302 \newcommand*\@writelinesonpageR}[1]{%
2303   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2304   \next}
2305
```

`\l@dcalc@maxoftwo` `\l@dcalc@maxoftwo{<num>}{<num>}{<count>}` sets `<count>` to the maximum of
`\l@dcalc@minoftwo` the two `<num>`.

Similarly `\l@dcalc@minoftwo{<num>}{<num>}{<count>}` sets `<count>` to the
 minimum of the two `<num>`.

```

2306 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2307   \ifnum #2>#1\relax
2308     #3=#2\relax
2309   \else
2310     #3=#1\relax
2311   \fi}
2312 \newcommand*{\l@dcalc@minoftwo}[3]{%
2313   \ifnum #2<#1\relax
2314     #3=#2\relax
2315   \else
2316     #3=#1\relax
2317   \fi}
2318

```

\ifl@dsamepage \checkpageL tests if the space and lines already taken on the page by text and foot-
 \l@dsamepagetrue notes is less than the constraints. If so, then \ifl@dpagetrue is set FALSE and
 \l@dsamepagefalse \ifl@dsamepage is set TRUE. If the page is spatially full then \ifl@dpagetrue
 \ifl@dpagetrue is set TRUE and \ifl@dsamepage is set FALSE. If it is not spatially full but
 \l@dpagetrue the maximum number of lines have been output then both \ifl@dpagetrue and
 \l@dpagetruefalse \ifl@dsamepage are set FALSE.

```

\checkpageL 2319 \newif\ifl@dsamepage
\checkpageR 2320 \l@dsamepagetrue
2321 \newif\ifl@dpagetrue
2322
2323 \newcommand*{\checkpageL}{%
2324   \l@dpagetrue
2325   \l@dsamepagetrue
2326   \check@goal
2327   \ifdim\pagetotal<\ledthegoal
2328     \ifnum\l@dpagetrue<\l@dsamepagetrue
2329       \else
2330         \l@dsamepagefalse
2331         \l@dpagetruefalse
2332       \fi
2333     \else
2334       \l@dsamepagefalse
2335       \l@dpagetrue
2336     \fi%
2337   \ifprint@last@after@pendL%
2338     \l@dpagetruefalse%
2339     \l@dsamepagefalse%
2340   \print@last@after@pendLfalse%
2341   \fi%
2342 }%
2343
2344 \newcommand*{\checkpageR}{%
2345   \l@dpagetrue
2346   \l@dsamepagetrue
2347   \check@goal

```

```

2348 \ifdim\pagetotal<\ledthegoal
2349 \ifnum\numpagelinesR<\l@dmminpagelines
2350 \else
2351 \l@dsamepagefalse
2352 \l@dpagfullfalse
2353 \fi
2354 \else
2355 \l@dsamepagefalse
2356 \l@dpagfulltrue
2357 \fi%
2358 \ifprint@last@after@pendR%
2359 \l@dpagfullfalse%
2360 \l@dsamepagefalse%
2361 \print@last@after@pendRfalse%
2362 \fi%
2363 }%
2364

```

`\checkpbL` `\checkpbL` and `\checkpbR` are called after each line is printed, and after the page is checked. These commands correct page breaks depending on `\ledpb` and `\lednopb`.

```

2365 \newcommand{\checkpbL}{
2366 \IfStrEq{\led@pb@setting}{after}{
2367 \xifinlistcs{\the\absline@num}{\l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{
2368 \xifinlistcs{\the\absline@num}{\l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{
2369 }{
2370 \IfStrEq{\led@pb@setting}{before}{
2371 \numdef{\next@absline}{\the\absline@num+1}
2372 \xifinlistcs{\next@absline}{\l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{
2373 \xifinlistcs{\next@absline}{\l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{
2374 }{
2375 }
2376 }
2377 \newcommand{\checkpbR}{
2378 \IfStrEq{\led@pb@setting}{after}{
2379 \xifinlistcs{\the\absline@numR}{\l@prev@pbR}{\l@dpagfulltrue\l@dsamepagefalse}{
2380 \xifinlistcs{\the\absline@numR}{\l@prev@nopbR}{\l@dpagfullfalse\l@dsamepagetrue}{
2381 }{
2382 \IfStrEq{\led@pb@setting}{before}{
2383 \numdef{\next@abslineR}{\the\absline@numR+1}
2384 \xifinlistcs{\next@abslineR}{\l@prev@pbR}{\l@dpagfulltrue\l@dsamepagefalse}{
2385 \xifinlistcs{\next@abslineR}{\l@prev@nopbR}{\l@dpagfullfalse\l@dsamepagetrue}{
2386 }{
2387 }

```

`\checkverseL` `\checkverseL` and `\checkverseR` are called after each line is printed. They prevent page break inside verse.

```

2388 \newcommand{\checkverseL}{
2389 \ifinstanzaL

```

```

2390 \iflednopbinverse
2391 \ifinserthangingsymbol
2392 \numgdef{\prev@abslineinverse}{\the\absline@num-1}
2393 \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineinverse}}{}
2394 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\prev@abslineinverse}\fi}{}
2395 \fi
2396 \fi
2397 \fi
2398 }
2399 \newcommand{\checkverseR}{
2400 \ifinstanzaR
2401 \iflednopbinverse
2402 \ifinserthangingsymbolR
2403 \numgdef{\prev@abslineinverse}{\the\absline@numR-1}
2404 \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineinverse}}{}
2405 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\prev@abslineinverse}\fi}{}
2406 \fi
2407 \fi
2408 \fi
2409 }

```

`\ledthegoal` `\ledthegoal` is the amount of space allowed to taken by text and footnotes on
`\goalfraction` a page before a forced pagebreak. This can be controlled via `\goalfraction`.
`\check@goal` `\ledthegoal` is calculated via `\check@goal`.

```

2410 \newdimen\ledthegoal
2411 \ifshiftedpstarts
2412 \newcommand*{\goalfraction}{0.95}
2413 \else
2414 \newcommand*{\goalfraction}{0.9}
2415 \fi
2416
2417 \newcommand*{\check@goal}{%
2418 \ledthegoal=\goalfraction\pagegoal}
2419

```

`\ifwrittenlinesL` Booleans for whether line data has been written to the section file.

```

\ifwrittenlinesL 2420 \newif\ifwrittenlinesL
2421 \newif\ifwrittenlinesR
2422

```

`\get@nextboxL` If the current box is not empty (i.e., still contains some lines) nothing is done.
`\get@nextboxR` Otherwise if and only if a synchronisation point is reached the next box is started.

```

2423 \newcommand*{\get@nextboxL}{%
2424 \ifvbox\namebox{1@dLcolrawbox\the\l@dpscl}% box is not empty

The current box is not empty; do nothing.

2425 \else%
box is empty

The box is empty. Check if enough lines (real and blank) have been output.

```

```

2426 \ifnum\usernamecount{l@dmxlinesinpar\the\l@dpscL}>\@donetotallinesL
2427 \parledgroup@notes@endL
2428 \else
    Sufficient lines have been output.
2429 \ifnum\usernamecount{l@dmxlinesinpar\the\l@dpscL}=\@donetotallinesL
2430 \parledgroup@notes@endL
2431 \fi
2432 \ifwrittenlinesL\else
    Write out the number of lines done, and set the boolean so this is only done once.
2433 \@writelinesinparL
2434 \writtenlinesLtrue
2435 \fi
2436 \ifnum\l@dnumpstartsL>\l@dpscL
    There are still unprocessed boxes. Recalculate the maximum number of lines
    needed, and move onto the next box (by incrementing \l@dpscL). If needed, restart
    the line numbering.
2437 \writtenlinesLfalse
2438 \ifbypstart@
2439 \global\line@num=0%
2440 \resetprevline%
2441 \fi
2442 % Add the content of the optional argument of the previous \cs{pend}.
2443 % \begin{macrocode}
2444 \csuse{after@pendL@the\l@dpscL}%
2445 \global\csundef{after@pendL@the\l@dpscL}%
    Check the number of lines
2446 \l@dcalc@maxoftwo{\the\usernamecount{l@dmxlinesinpar\the\l@dpscL}}%
2447 {\the\@donetotallinesL}%
2448 {\usernamecount{l@dmxlinesinpar\the\l@dpscL}}%
2449 \global\@donetotallinesL \z@
    Go to the next pstart
2450 \global\advance\l@dpscL \@ne
2451 \global\pstartnumtrue%
2452 \restore@pstartL@pc%
    Add notes of parallel ledgroup.
2453 \parledgroup@notes@endL
2454 \parledgroup@correction@notes@spacing@final{L}
2455 \else
    Add the content of the optional argument of the last \pend.
2456 \print@last@after@pendLtrue%
2457 \fi
2458 \fi
2459 \fi}

```



```

2460 \newcommand*{\get@nextboxR}{%
2461   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2462   \else% box is empty
2463     \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2464       \parledgroup@notes@endR
2465     \else
2466       \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2467         \parledgroup@notes@endR
2468       \fi
2469     \ifwrittenlinesR\else
2470       \@writelinesinparR
2471       \writtenlinesRtrue
2472     \fi
2473     \ifnum\l@dnumpestartsR>\l@dpscR
2474       \writtenlinesRfalse
2475       \ifbypstart@R
2476         \global\line@numR=0%
2477         \resetprevline@%
2478       \fi
2479       \csuse{after@pendR@the\l@dpscR}%
2480       \global\csundef{after@pendR@the\l@dpscR}%
2481       \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2482         {\the\@donetotallinesR}%
2483         {\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}}%
2484       \global\@donetotallinesR \z@
2485       \global\advance\l@dpscR \@ne
2486       \global\pstartnumRtrue%
2487       \restore@pstartR@pc%
2488       \parledgroup@notes@endR
2489       \parledgroup@correction@notes@spacing@final{R}
2490     \else
2491       \print@last@after@pendRtrue%
2492     \fi
2493   \fi
2494 \fi}
2495

```

29 Sections' titles' commands

`\eledsectnotoc` `\eledsectnotoc` just saves its content `\@eledsectnotoc`, which will be tested where sectioning commands will be printed.

```

2496 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2497 \eledsectnotoc{R}

```

`\eledsectmark` `\eledsectmark` just saves its content `\@eledsectmark`, which will be tested where sectioning commands will be printed.

```

2498 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2499 \eledsectmark{L}

```

`\eledsection@correcting@skip` Because the vertical correction needed after inserting a title in parallel depends whether we are in parallel columns or parallel pages, we stock its length in `\eledsection@correcting@skip`.

2500 `\newskip\eledsection@correcting@skip`

`\eled@sectioningR@out` We save the sectioning commands of the right side in the `\eled@sectioningR@out` file.

2501 `\newwrite\eled@sectioningR@out`

30 Page break/no page break, depending on the specific line

We need to adapt the macro of the homonym section of `eledmac` to `eledpar`.

`\prev@pbR` The `\l@prev@pbR` macro is a etoolbox list, which contains the lines in which page breaks occur (before or after). The `\l@prev@nopbR` macro is a etoolbox list, which contains the lines in which NO page breaks occur (before or after).

2502 `\def\l@prev@pbR{}`
 2503 `\def\l@prev@nopbR{}`

`\ledpbR` The `\ledpbR` macro writes the call to `\led@pbR` in line-list file. The `\ledpbnumR` macro writes the call to `\led@pbnumR` in line-list file. The `\lednopbR` macro writes the call to `\led@nopbR` in line-list file. The `\lednopbnumR` macro writes the call to `\led@nopbnumR` in line-list file.

2504 `\newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}`
 2505 `\newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}`
 2506 `\newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}`
 2507 `\newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\led@nopbnumR{#1}}}`

`\led@pbR` The `\led@pbR` add the absolute line number in the `\prev@pbR` list. The `\led@pbnumR` add the argument in the `\prev@pbR` list. The `\led@nopbR` add the absolute line number in the `\prev@nopbR` list. The `\led@nopbnumR` add the argument in the `\prev@nopbR` list.

2508 `\newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}`
 2509 `\newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}`
 2510 `\newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}`
 2511 `\newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}`

31 Parallel ledgroup

`\parledgroup@` The marks `\parledgroup` contains information about the beginnings and endings of notes in a parallel ledgroup. `\parledgroupseries@` contains the footnote series.

`\parledgrouptype@` `\parledgroupseries` contains the type of the footnote: critical (Xfootnote) or familiar (footnoteX).

2512 `\newmarks\parledgroup@`

```

2513 \newmarks\parledgroup@series
2514 \newmarks\parledgroup@type

```

`\parledgroup@notes@startL` `\parledgroup@notes@startL` and `\parledgroup@notes@startR` are used to mark the beginning of a note series in a parallel ledgroup.

```

2515 \newcommand{\parledgroup@notes@startL}{%
2516   \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscL}>0%
2517     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmarks\parledg
2518     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmarks\parledg
2519   \fi%
2520   \global\ledgroupnotesL=true%
2521   \insert@noterule@ledgroup{L}%
2522 }
2523 \newcommand{\parledgroup@notes@startR}{%
2524   \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}>0%
2525     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmarks\parledg
2526     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmarks\parledg
2527   \fi%
2528   \global\ledgroupnotesR=true%
2529   \insert@noterule@ledgroup{R}%
2530 }

```

`\parledgroup@notes@startL` `\parledgroup@notes@endL` and `\parledgroup@notes@endR` are used to mark the end of a note series in a parallel ledgroup.

```

2531 \newcommand{\parledgroup@notes@endL}{%
2532   \global\ledgroupnotesL=false%
2533 }
2534 \newcommand{\parledgroup@notes@endR}{%
2535   \global\ledgroupnotesR=false%
2536 }

```

`\insert@noterule@ledgroup` A `\vskip` is not used when the boxes are constructed. So we insert it before ledgroup note series when paralling lines are constructed. This is the goal of `\insert@noterule@ledgroup`

```

2537 \newcommand{\insert@noterule@ledgroup}[1]{
2538   \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2539     \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2540       \csuse{ifledgroupnotes#1@}
2541       \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2542       \csuse{\splitbotmarks\parledgroup@series footnoterule}
2543     \fi
2544   }
2545   {}
2546   \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2547     \csuse{ifledgroupnotes#1@}
2548     \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2549     \csuse{footnoterule\splitbotmarks\parledgroup@series}
2550     \fi
2551   }{}

```

```

2552     }
2553     {}
2554 }

```

`\parledgroupnotespacing` `\parledgroupnotespacing` can be redefined by the user to change the interline spacing of ledgroup notes.

```

2555 \newcommand{\parledgroupnotespacing}{}

```

`\parledgroup@notespacing@correction` `\parledgroup@notespacing@correction` is the difference between a normal line skip and a line skip in a note. It's set by `\parledgroup@notespacing@set@correction`, called at the beginning of `\Pages`.

```

2556 \dimdef{\parledgroup@notespacing@correction}{0pt}
2557 \newcommand{\parledgroup@notespacing@set@correction}{%
2558   {\notefontsetup\parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
2559   \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing}%
2560 }

```

`\parledgroup@correction@notespacing@init` `\parledgroup@correction@notespacing@init` sets the value of accumulated corrections of note spacing to 0 pt. It's called at the beginning of each pages AND at the end of each ledgroup.

```

2561 \newcommand{\parledgroup@correction@notespacing@init}{
2562   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2563   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2564 }
2565 \parledgroup@correction@notespacing@init

```

`\parledgroup@correction@notespacing@final` `\parledgroup@correction@notespacing@final` adds the total space deleted because of correction for notes, in a parallel ledgroup. It also adds the space needed by the other side spaces between note rules and notes. It's called after the print of each pstart/pend.

```

2566 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2567   \ifparledgroup
2568     \vspace{\parledgroup@notespacing@correction@accumulated}
2569     \parledgroup@correction@notespacing@init%
2570     \ifstrequal{#1}{L}{
2571       \numdef{\@checking}{\the\l@dpscL-1}
2572     }{
2573       \numdef{\@checking}{\the\l@dpscR-1}
2574     }
2575     \dimdef{\@beforenotes@current@diff}{\csuse{\parledgroup@beforenotes@\@checking L}-\cs
2576     \ifstrequal{#1}{L}%
2577       {% Left
2578         \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\@beforenotes@current@dif
2579       }%
2580       {% Right
2581         \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\@beforenotes@current@diff}}
2582       }%
2583     \fi
2584 }

```

`\parledgroup@correction@notespacing` `\parledgroup@correction@notespacing` is used before each printed line. If it's a line of notes in parallel ledgroup, the space `\parledgroup@notespacing@correction` is decreased, to make interline space correct. The decreased space is added to `\parledgroup@notespacing@correction@accumulated` and `\parledgroup@notespacing@correction@modulo`. If `\parledgroup@notespacing@correction@modulo` is equal or greater than `\baselineskip`:

- It is decreased by `\baselineskip`.
- The total of line number in the current page is decreased by one.

For example, suppose an normal interline of 24 pt and interline for note of 12 pt. That means that the two lines of notes take the place of one normal line. For every two lines of notes, the line total for the current place is decreased by one.

```

2585 {}
2586 \newcommand{\parledgroup@correction@notespacing}[1]{%
2587     \csuse{ifledgroupnotes#1@}%
2588     \vspace{-\parledgroup@notespacing@correction}%
2589     \dimdef{\parledgroup@notespacing@correction@accumulated}{\parledgroup@notespacing@correction@ac
2590     \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo+
2591     \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip}{\advance\numpagelinesL
2592     \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo-
2593     }% mean greater than equal
2594     \fi%
2595 }
```

`\parledgroup@beforenotesL` `\parledgroup@beforenotesL` and `\parledgroup@beforenotesR` store the total of space before notes in the current parallel ledgroup.

```

2596 \dimdef\parledgroup@beforenotesL{0pt}
2597 \dimdef\parledgroup@beforenotesR{0pt}
```

`\parledgroup@beforenotes@save` The macro `\parledgroup@beforenotes@save` dumps the space before notes of the current parallel ledgroup in a macro named with the current `pstart` number.

```

2598 \newcommand{\parledgroup@beforenotes@save}[1]{
2599     \ifparledgroup
2600         \csdimgdef{\parledgroup@beforenotes@the\csuse{lednumstarts#1}#1}{\csuse{\parledgroup@beforenotes
2601         \csdimgdef{\parledgroup@beforenotes#1}{0pt}
2602     \fi
2603 }
```

32 The End

`i/code`

Appendix A Some things to do when changing version

Appendix A.1 Migration to eledpar 1.4.3

Version 1.4.3 corrects a bug added in version 0.12, which made hanging verse automatically flush right, despite the given value of the first element of the `\setstanzaindent` command.

If, however, you want to return to automatic flush-right margins for verses with hanging indents, you have to redefine the `\hangingsymbol` command.

```
\renewcommand{\hangingsymbol}{\protect\hfill}
```

See the two following examples:

With standard `\hangingsymbol`:

A very long verse should be sometime hanged. The position of the hanging verse is fixed.

With the modification of `\hangingsymbol`:

A very long verse should sometimes be hanging. And we can see that an hanging verse is flush right.

References

- [LW90] John Lavagnino and Dominik Wujastyk. ‘An overview of EDMAC: a PLAIN TeX format for critical editions’. *TUGboat*, **11**, 4, pp. 623–643, November 1990. (Code available from CTAN in `macros/plain/contrib/edmac`)
- [Wil02] Peter Wilson. *The memoir class for configurable typesetting*. November 2002. (Available from CTAN in `macros/latex/contrib/memoir`)
- [Wil04] Peter Wilson and Maïeul Rouquette. *eledmac A presumptuous attempt to port EDMAC, TABMAC and EDSTANZA to LaTeX*. December 2004. (Available from CTAN in `macros/latex/contrib/eledmac`)

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	<code>\@afterindentfalse</code> 767
<code>\&</code> 1662, 1663, 1667, 1683, 1706	<code>\@arabic</code> 216, 217, 815, 817
<code>\@adv</code> <u>364</u> , 624, 625	<code>\@astanza@line</code> 1682, 1685, <u>1693</u>

- \@auxout 1771
 - \@beforenotes@current@diff
..... 2575, 2578, 2581
 - \@chapter 768
 - \@checking 2571, 2573, 2575
 - \@cs@linesinparL 2103, 2273
 - \@cs@linesinparR 2103, 2273
 - \@cs@linesonpageL .. 2111, 2211, 2286
 - \@cs@linesonpageR .. 2111, 2211, 2286
 - \@currentlabel 861, 908
 - \@donereallinesL
..... 969, 999, 1839, 1841, 2119
 - \@donereallinesR
..... 969, 1086, 1844, 1846, 2121
 - \@donetotallinesL 969, 1000,
1003, 2120, 2426, 2429, 2447, 2449
 - \@donetotallinesR 969, 1087,
1090, 2122, 2463, 2466, 2482, 2484
 - \@eled@sectioningfalse 1037
 - \@eled@sectioningtrue 1028
 - \@eledsectionL . 1848, 1882, 1898, 1899
 - \@eledsectionR . 1848, 1893, 1903, 1904
 - \@eledsectmark . 1025, 1878, 1889, 2498
 - \@eledsectnotoc 1024, 2129, 2170, 2496
 - \@gobble 553
 - \@gobbletwo 558
 - \@insertR 1392–1394, 1407–1409
 - \@l@dttempcnta 437,
439, 441, 442, 446, 448, 450,
451, 1144, 1183, 1184, 1186,
1188, 1191, 1192, 1209–1213,
1215, 1222, 1227, 1231, 1239,
1244, 1248, 1281, 1284, 1286, 1290
 - \@l@dttempcntb 172, 174, 176,
1174, 1175, 1222, 1227, 1231,
1239, 1244, 1248, 1273, 1277,
1290, 1298–1300, 1302, 1323–
1325, 1327, 1344–1346, 1348,
1537, 1538, 1566–1568, 1570,
1734–1738, 1742–1745, 1749–1752
 - \@lab 557, 1524
 - \@lock 987, 1124
 - \@lockR . 50, 308, 310, 312, 325, 471,
487, 488, 490, 491, 519, 520,
522, 1073, 1106, 1150, 1152,
1153, 1155, 1236, 1253, 1255, 1257
 - \@lopL 582, 2300
 - \@lopR 553, 582, 2303
 - \@nl 286, 607, 609
 - \@nl@reg 335
 - \@nl@regR 286
 - \@nobreakfalse 823, 871, 1027
 - \@nobreaktrue . 821, 825, 869, 873, 1027
 - \@nobptrue 1964, 1965, 1970, 1971
 - \@oldnobreak 821, 823, 869, 871, 926, 948
 - \@pbtrue 1962, 1963, 1968, 1969
 - \@pend 573, 1839
 - \@pendR 573, 1844
 - \@pstartsfalse 1812
 - \@pstartstrue 1812
 - \@ref 543
 - \@ref@reg 571
 - \@schapter 768
 - \@series 713, 2241
 - \@set 396, 631, 632, 1917, 1921
 - \@startstanza 780, 781, 803, 804
 - \@stopstanza 1683, 1688
 - \@sw 558
 - \@tag 1591, 1593, 1597
 - \@temp 730, 731, 736, 737
 - \@templ@ed 1556, 1558
 - \@templ@n 1557, 1558
 - \@tmp 690, 692, 693, 696–698
 - \@tmpa 691, 692, 697, 700
 - \@writelinesinparL .. 1837, 1913, 2433
 - \@writelinesinparR .. 1837, 1914, 2470
 - \@writelinesonpageL . 2159, 2161, 2299
 - \@writelinesonpageR . 2200, 2202, 2299
 - \@xloop 1405
- A**
- \absline@num
360, 430, 444, 463, 1115, 1960,
1968, 1970, 2367, 2368, 2371, 2392
 - \absline@numR 45, 233, 288,
291, 294, 427, 435, 456, 475,
509, 537, 548, 1097, 1136, 1137,
1174, 1391, 1961, 1969, 1971,
2379, 2380, 2383, 2403, 2508, 2510
 - \actionlines@list
..... 278, 281, 430, 444, 463
 - \actionlines@listR
..... 237, 254, 270, 273, 427,
435, 456, 475, 509, 537, 1196, 1199
 - \actions@list . 282, 431, 451, 465, 467
 - \actions@listR
..... 237, 255, 274, 428, 442, 458,
460, 477, 486, 511, 518, 538, 1200
 - \add@inserts 996, 1010
 - \add@inserts@nextR 1380

- \add@insertsR 1082, 1380
 \add@penaltiesL 998, 1401
 \add@penaltiesR 1085, 1401
 \addtocontents 1772–1774
 \addtocounter 929, 951, 1023
 \advanceline 623, 654
 \affixline@num 994
 \affixline@numR 1080, 1206
 \affixpstart@numL 1008, 1313
 \affixpstart@numR 1313
 \affixside@note 996, 1010
 \affixside@noteR 1082, 1543
 \aftercolumnseparator . . 4, 1944, 1985
 \appto 1549, 1550
 \araw@textfalse 1824
 \araw@texttrue 1824
 astanza (environment) 10, 1665
 \at@begin@pairs 744, 749, 750
 \at@every@pend 933, 955
 \at@every@pstart 864, 911
 \AtBeginDocument . . . 1520, 1756, 1782
 \AtBeginPairs 4, 749
- B**
- \ballast@count 1134, 1139
 \bbl@main@language 1798, 1799
 \bbl@set@language 1789, 1790
 \beforecolumnseparator
 4, 1938, 1941, 1947, 1985
 \beginnumbering 8, 791, 833
 \beginnumberingR . . . 36, 117, 791, 881
 \bfseries 815, 817
 \boolfalse 1444, 1488
 \booltrue 1447, 1491
 \bypage@Rfalse 137, 152, 157
 \bypage@Rtrue 137, 147
 \bypstart@Rfalse 137, 148, 158
 \bypstart@Rtrue 137, 153
- C**
- \c@ballast 1139
 \c@chapter 98
 \c@chapterR 98
 \c@firstlinenumR 181, 1279
 \c@firstsublinenumR 185, 1274
 \c@linenumincrementR 181, 1279
 \c@page 607, 609, 1432, 1434,
 1476, 1478, 2248, 2254, 2258, 2267
 \c@pstartL 719, 731, 815
 \c@pstartR 724, 737, 817
 \c@section 99
 \c@sectionR 99
 \c@sublinenumincrementR . . 185, 1274
 \c@subsection 100
 \c@subsectionR 100
 \c@subsubsection 101
 \c@subsubsectionR 101
 \ch@ck@l@ckR 1206
 \ch@cksub@l@ckR 1206
 \ch@cksub@lockR 1275
 \chapter 753, 754, 763
 \chapterinpages 740, 754, 765
 \chardef 1662
 \check@goal 2326, 2347, 2410
 \check@pstarts
 1812, 1864, 1915, 2097, 2105, 2113
 \checkpageL 2126, 2154, 2319
 \checkpageR 2166, 2195, 2319
 \checkpb@columns . . . 1911, 1952, 1956
 \checkpbL 2156, 2365
 \checkpbR 2197, 2365
 \checkraw@text
 1824, 1872, 1908, 2123, 2207
 \checkverseL 1909, 2155, 2388
 \checkverseR 1910, 2196, 2388
 \cleardoublepage 2261
 \clearl@dleftpage 2165, 2255
 \clearl@drightpage 2206, 2255
 \cleartoevenpage 2246, 2270
 \cleartol@devenpage 2084, 2246
 \closeout 89, 595, 602
 \columnrulewidth 4, 1976
 \Columns 3, 1850
 \columns@position . . 1896, 1906, 1981
 \columnseparator 4, 1943, 1976
 \columnspan 4, 1981
 \correct@footinsX@box 1425
 \correct@Xfootins@box 1425
 \correcthangingL 1019, 1643
 \correcthangingR 1643
 \count 1440, 1448, 1484, 1492
 \countLline 964, 976
 \countRline 964, 1062
 \critext 659
 \cs 1454, 1953, 2442
 \csdingdef 2600, 2601
 \csgdef 681, 682, 864,
 865, 911, 912, 933, 934, 955, 956
 \cslet 693, 698, 720, 725

\csundef 1038, 1881, 1892,
2131, 2152, 2172, 2193, 2445, 2480
\csuse 688, 691, 701, 702, 1035, 1427,
1428, 1438–1443, 1448, 1449,
1451, 1452, 1460, 1462, 1463,
1465, 1471, 1472, 1482–1487,
1492, 1493, 1495, 1496, 1503,
1505, 1506, 1508, 1612, 1623,
1879, 1890, 2130, 2151, 2171,
2192, 2231, 2232, 2444, 2479,
2517, 2518, 2525, 2526, 2540–
2542, 2547–2549, 2575, 2587, 2600

D

\DeclareOption 9–12
\def@tempb 155
\dimdef 2556, 2562, 2563,
2575, 2589, 2590, 2592, 2596, 2597
\dimen 610,
611, 615–617, 621, 1443, 1452,
1487, 1496, 2141–2143, 2182–2184
\dimexpr 1463, 1464, 1506, 1507
\dingdef 2558, 2559
\divide 1211, 2000, 2021, 2037, 2049, 2064
\do@actions 1116
\do@actions@fixedcodeR 1143
\do@actions@nextR 1143
\do@actionsR 1098, 1143
\do@ballast 1117
\do@ballastR 1099, 1134
\do@insidelineLhook . 1013, 1051, 1054
\do@insidelineRhook 1052, 1054
\do@lineL 974, 1875, 2132
\do@lineLhook 980, 1049, 1054
\do@lineR 1059, 1886, 2173
\do@lineRhook 1050, 1054, 1066
\do@lockoff 506
\do@lockoffL 530
\do@lockoffR 506
\do@lockon 471
\do@lockonL 503
\do@lockonR 471
\doinsidelinehook 2026, 2054
\doinsidelineLhook 1049
\doinsidelineRhook 1049
\dolineLhook 1049
\dolineRhook 1049
\dolistloop 713, 1554, 1574, 1581, 2241
\dummy@ref 552
\dump@pstartL@pc 718, 927

\dump@pstartR@pc 718, 949

E

\edfont@info 663, 666, 672, 675
\edlabel 1514
\edtext 659
\eled@sectioningR@out .. 64, 89, 2501
\eled@sections@@ 995, 1027, 1876, 2133
\eled@sectionsR@@ 61, 1081, 1887, 2174
\eledpar@error 19, 21, 24, 27, 29
\eledsection@correcting@skip ...
..... 1046, 1852, 2075, 2500
\eledsectmark 13, 2498
\eledsectnotoc 13, 2496
\empty 74, 77, 270, 278,
661, 670, 729, 735, 841, 889,
1196, 1278, 1286, 1382–1384,
1395, 1406, 2274, 2280, 2287, 2293
\endashchar 1417
\endgraf 922, 944, 1858, 2087
\endline@num 561, 567
\endlock 643, 1671, 1679, 1689
\endnumbering 8, 67, 121, 792
\endnumberingR 39, 67, 104, 116, 129, 792
\endpage@num 560, 567
\endstanzaextra 1691
\endsub 610
\endsubline@num 562, 568
\enlargethispage 1973
environments:
 astanza 10, 1665
 Leftside 7, 772
 pages 5, 740
 pairs 3, 740
 Rightside 7, 789
\expandonce 1593, 1597, 1613, 1624
\extensionchars 56, 110, 126, 134

F

\f@x@l@cksR 1206
\finish@Pages@notes 2226, 2229
\first@linenum@out@Rfalse .. 590, 596
\first@linenum@out@Rtrue 590
\firstlinenum 7, 190
\firstlinenum* 7, 190
\firstsublinenum 7, 190
\firstsublinenum* 7, 190
\fix@page 331, 338
\flag@end 610
\flag@start 610

```

\ifbypstart@R ... 137, 579, 1920, 2475
\ifcseempty ..... 1426, 1470
\ifcsstring .... 1432, 1434, 1476, 1478
\ifdefstring ..... 1024, 1025,
                    1878, 1889, 1896, 1906, 2129, 2170
\ifdim ..... 611,
                615, 617, 621, 1938, 1944, 2138,
                2179, 2250, 2256, 2265, 2327, 2348

```

\ifdimgreater	2578, 2581
\ifdimless	2591
\iffirst@linenum@out@R	590, 594
\ifhbox	1898, 1903
\ifinserthangingsymbol	
	1632, 1646, 2391

\ifinserthangingsymbolR	
	1630, 1638, 1656, 2402
\ifistanzaL	770, <u>770</u> , 1633, 1645, 2389	
\ifistanzaR	<u>770</u> , 771, 1639, 1655, 2400	
\ifl@d@dash	1417
\ifl@d@elin	1419, 1420
\ifl@d@esl	1420
\ifl@d@pnum	1414, 1418
\ifl@d@ssub	1416
\ifl@b@full	2158, 2160, 2311

\ifl@pagerull	2158, 2199, <u>2319</u>
\ifl@dpadding	<u>14</u> , 1644, 1654
\ifl@dpairing	<u>14</u> , 71
\ifl@dsamelang	<u>1767</u>
\ifl@dsamepage	2128, 2168, <u>2319</u>
\ifl@dskipnumber	1269
\ifl@dusedbabel	<u>1765</u>
\ifl@elpstart	861, 908
\ifledgroupnotesL@	1118
\ifledgroupnotesR@	1100, 1268
\iflednopbinverse	2390, 2401
\ifledplinenum	1415

\filedRcol . 14, 173, 195, 199, 203,
207, 251, 268, 332, 341, 366,
380, 397, 414, 426, 434, 455,
500, 527, 536, 545, 612, 618,
624, 631, 639, 644, 648, 653,
660, 1525, 1592, 1605, 1793, 1805

\ifluatex	
. 826, 874, 1014, 1030, 1606, 1617	
\ifnoteschanged@	81
\ifnumberedpar@	

```

... 835, 883, 918, 940, 1590, 1604
\ifnumbering ..... 142, 831, 915
\ifnumberingR ... 37, 68, 106, 879, 937
\ifnumberline ..... 1100, 1118, 1268

```

- \ifnumberstart 720,
 - 725, 852, 900, 928, 950, 975, 1060
 - \ifnumequal 1547
 - \ifnumgreater 1555, 1575, 1582
 - \ifnumodd 1432, 1434, 1476, 1478
 - \ifodd 1302, 1327,
 - 1348, 1570, 2248, 2254, 2258, 2267
 - \ifparledgroup 2567, 2599
 - \ifprint@last@after@pendL
..... 959, 2150, 2337
 - \ifprint@last@after@pendR
..... 959, 2191, 2358
 - \ifpst@rtedL 32, 839
 - \ifpst@rtedR 32, 887
 - \ifpstartnum 1358, 1363
 - \ifpstartnumR 1313
 - \ifshiftedpstarts 6, 2137, 2178, 2411
 - \ifsidepstartnum 853, 901, 1315, 1336
 - \ifstrempty 863, 910, 932, 954
 - \IfStrEq 984, 1070, 1959,
 - 1967, 2366, 2370, 2378, 2382,
 - 2393, 2394, 2404, 2405, 2517,
 - 2518, 2525, 2526, 2538, 2539, 2546
 - \ifstrequal 2570, 2576
 - \ifsublines@ 227,
 - 320, 365, 398, 405, 436, 445,
 - 457, 464, 476, 510, 566, 568,
 - 1101, 1119, 1185, 1272, 1527, 1531
 - \ifvbox 977, 1063, 1828, 1831, 2424, 2461
 - \ifvoid 2231, 2232
 - \ifwidthliketwocolumns 12
 - \ifwrittenlinesL 2420, 2432
 - \ifwrittenlinesR 2421, 2469
 - \init@series@eledpar 711
 - \initnumbering@sectcmd 743, 757
 - \initnumbering@sectcountR
..... 60, 93, 113, 2169
 - \InputIfFileExists 63
 - \insert@count 542, 1599, 1626
 - \insert@countR 543, 1595, 1615
 - \insert@noterule@ledgroup
..... 2521, 2529, 2537
 - \inserthangingsymbolfalse 990
 - \inserthangingsymbolL ... 1018, 1630
 - \inserthangingsymbolR 1630
 - \inserthangingsymbolRfalse 1076
 - \inserthangingsymbolRtrue 1074
 - \inserthangingsymboltrue 988
 - \insertlines@listR
.... 74, 237, 253, 548, 1384, 1388
 - \inserts@list 840, 1598, 1625
 - \inserts@listR 888, 1379,
1382, 1392, 1406, 1407, 1594, 1614
 - \instanzaLfalse 1934, 2223
 - \instanzaLtrue 781
 - \instanzaRfalse 1935, 2224
 - \instanzaRtrue 804
 - \itemcount@ 1545, 1547,
1552, 1555, 1573, 1575, 1580, 1582
- L**
- \l@d@nums 663, 666, 672, 675
 - \l@d@set 413, 639, 640
 - \l@dbbl@set@language 1767, 1790
 - \l@dbfnote 1589
 - \l@dc@maxchunks 846, 848,
894, 896, 1724, 1734, 1742, 1749
 - \l@dcalc@maxoftwo
..... 2103, 2306, 2446, 2481
 - \l@dcalc@minoftwo .. 2111, 2211, 2306
 - \l@dcalcnun 1206
 - \l@dchecklang 1767
 - \l@dchset@num 287, 290, 413
 - \l@dcsnotetext 1554, 1557
 - \l@dcsnotetext@l 1557, 1574
 - \l@dcsnotetext@r 1557, 1581
 - \l@demptyd@ta 981, 1067
 - \l@dend@stuff 57, 111, 127, 135
 - \l@dgetline@margin 171
 - \l@dgetsidenote@margin 1536
 - \l@dld@ta 1009,
1293, 1305, 1318, 1330, 1339, 1351
 - \l@dleftbox 961,
992, 1002, 1897, 2138, 2139, 2147
 - \l@dlinenumR 219
 - \l@dlsn@te 1011
 - \l@dmake@labelsR 1514
 - \l@dminpagelines
.... 2069, 2112, 2212, 2328, 2349
 - \l@dnumpstartsL 720, 845,
846, 848, 850, 864, 865, 933,
934, 1728, 1760, 1815, 1853,
1854, 1930, 2079, 2080, 2221, 2436
 - \l@dnumpstartsR 41, 725, 893,
894, 896, 898, 911, 912, 955,
956, 1728, 1761, 1818, 1853,
1854, 1931, 2079, 2080, 2222, 2473
 - \l@doldbbl@set@language 1789
 - \l@doldselectlanguage 1788, 1792, 1797

- \l@dpagfullfalse 2319, 2368, 2373, 2380, 2385
- \l@dpagfulltrue 2319, 2367, 2372, 2379, 2384
- \l@dpagingfalse 742, 762
- \l@dpagingtrue 756
- \l@dpairingfalse 746, 761
- \l@dpairingtrue 741, 755
- \l@dpprintingcolumnfalse 1932
- \l@dpprintingcolumnstrue 1851
- \l@dpprintingpagesfalse 2225
- \l@dpprintingpagetrue 2074
- \l@dpscL 975,
977, 982, 995, 1026, 1035, 1038,
1730, 1762, 1815, 1828, 1862,
1868, 1876, 1879, 1881, 1928,
2091, 2099, 2104, 2107, 2115,
2130, 2131, 2133, 2151, 2152,
2219, 2424, 2426, 2429, 2436,
2444–2446, 2448, 2450, 2516, 2571
- \l@dpscR 1060, 1063, 1068,
1081, 1731, 1763, 1818, 1831,
1863, 1869, 1887, 1890, 1892,
1929, 2092, 2100, 2108, 2116,
2171, 2172, 2174, 2192, 2193,
2220, 2461, 2463, 2466, 2473,
2479–2481, 2483, 2485, 2524, 2573
- \l@drd@ta 1019,
1295, 1303, 1320, 1328, 1341, 1349
- \l@drightbox 961,
1078, 1089, 1902, 2179, 2180, 2188
- \l@drsn@te 1020
- \l@dsamepagefalse
.... 2319, 2367, 2372, 2379, 2384
- \l@dsamepagetrue
.... 2319, 2368, 2373, 2380, 2385
- \l@dsetupmaxlinecounts .. 1741, 1758
- \l@dsetuprawboxes 1733, 1757
- \l@dskipnumberfalse 1270
- \l@dskipnumbertrue 1166
- \l@dunhbox@line 1019, 1041, 1044
- \l@dusedbabelfalse 1765, 1785
- \l@dusedbabeltrue 1765, 1787
- \l@duselanguage
.... 1777, 1874, 1885, 2127, 2167
- \l@dzero maxlinecounts ... 1741, 1759
- \l@dzero penalties 921, 943, 1857, 2086
- \l@luatexbodydir@L 829, 1033
- \l@luatexbodydir@R 877
- \l@luatexpardir@L 828, 1032
- \l@luatexpardir@R 876
- \l@luatextextdir@L ... 827, 1015, 1031
- \l@luatextextdir@R 875
- \l@prev@nopbR ... 48, 2503, 2510, 2511
- \l@prev@pbR 47, 2502, 2508, 2509
- \l@pscL 1730
- \l@pscR 1730
- \labelref@list 1532
- \labelref@listR 1512, 1528
- \language name .. 1768, 1769, 1771–1774
- \last@page@num 352, 358
- \last@page@numR 338
- \lastbox 985, 1071
- \lastskip 610, 616
- \lcolwidth 4,
5, 14, 758, 859, 993, 1002, 1882,
1996, 2017, 2034, 2045, 2061, 2082
- \led@err@BadLeftRightPstarts ...
..... 23, 1854, 2080
- \led@err@LeftOnRightPage ... 26, 2259
- \led@err@LineationInNumbered ... 143
- \led@err@ManyLeftnotes 1575
- \led@err@ManyRightnotes 1582
- \led@err@ManySidenotes 1555
- \led@err@NumberingNotStarted ... 85
- \led@err@numberingShouldHaveStarted
..... 115
- \led@err@NumberingStarted 38
- \led@err@PendNoPstart 919, 941
- \led@err@PendNotNumbered ... 916, 938
- \led@err@PstartInPstart ... 836, 884
- \led@err@PstartNotNumbered . 832, 880
- \led@err@RightOnLeftPage ... 26, 2268
- \led@err@TooManyPstarts 20, 847, 895
- \led@mess@NotesChanged 82
- \led@mess@SectionContinued
..... 109, 125, 133
- \led@nopbnumR 2507, 2508
- \led@nopbR 2506, 2508
- \led@pb@setting
..... 1959, 1967, 2366, 2370,
2378, 2382, 2393, 2394, 2404, 2405
- \led@pbnumR 2505, 2508
- \led@pbR 2504, 2508
- \led@warn@BadAction 1168
- \led@warn@BadAdvancelineLine 383, 389
- \led@warn@BadAdvancelineSubline .
..... 369, 375
- \led@warn@BadLineation 160
- \led@warn@BadSetline 629

- \led@warn@BadSetlinenum 637
 - \led@warn@DuplicateLabel 1516
 - \ledgroupnotesL@false 2532
 - \ledgroupnotesL@true 2520
 - \ledgroupnotesR@false 2535
 - \ledgroupnotesR@true 2528
 - \ledllfill 1012
 - \lednopb 800
 - \lednopbnum 2393, 2504
 - \lednopbnumR 2404, 2504
 - \lednopbR 800, 2506
 - \ledpb 799
 - \ledpbnum 2394
 - \ledpbnumR 2405, 2504
 - \ledpbR 799, 2504
 - \ledRcol@false 1092
 - \ledRcol@true 1061
 - \ledRcolfalse 773, 806
 - \ledRcoltrue 790
 - \ledrlfill 1019, 2005, 2027
 - \ledsavedprintlines 9, 1412
 - \ledsectnomark 1025
 - \ledsectnotoc 1024, 2129, 2170
 - \ledstrutL 2139, 2147, 2243
 - \ledstrutR 2180, 2188, 2243
 - \ledthegoal 2327, 2348, 2410
 - \leftlinenumR 219, 1293, 1305
 - \leftpstartnumL 1313
 - \leftpstartnumR 1313
 - Leftside (environment) 7, 772
 - \Leftsidehook 779, 784
 - \Leftsidehookend 783, 784
 - \letcs 690, 696, 699, 975, 1060
 - \line@list 670, 674
 - \line@list@stuff 126
 - \line@list@stuffR .. 56, 110, 134, 592
 - \line@listR . 77, 237, 252, 568, 661, 665
 - \line@margin 176, 1323
 - \line@marginR 169, 1298, 1344
 - \line@num . 355, 387, 388, 390, 408,
419, 420, 448, 575, 1125, 1530, 2439
 - \line@numR 49,
226, 233, 292, 326, 345, 381,
382, 384, 401, 415, 416, 439,
561, 565, 579, 1107, 1179, 1188,
1277, 1279, 1281, 1282, 1526, 2476
 - \lineation 165, 166, 801
 - \lineation* 8, 165
 - \lineationR 8, 141, 167, 801
 - \linenum@out ... 607, 613, 619, 625,
632, 640, 645, 649, 1839, 1917, 2300
 - \linenum@outR 589,
595, 597, 602, 603, 609, 612,
618, 624, 631, 639, 644, 648,
653, 1844, 1921, 2303, 2504–2507
 - \linenumberlist 1278, 1282
 - \linenumincrement 7, 190
 - \linenumincrement* 7, 190
 - \linenummargin 169
 - \linenumr@p ... 1415, 1419, 1526, 1530
 - \linenumrepR 216, 226
 - \linenumsep
. 221, 223, 1360, 1363, 1372, 1375
 - \linesinpar@listL
..... 244, 262, 576, 2274, 2277
 - \linesinpar@listR
..... 244, 256, 580, 2280, 2283
 - \lineskip 1464, 1507
 - \linesonpage@listL 263, 584, 2287, 2290
 - \linesonpage@listR 257, 587, 2293, 2296
 - \list@clear
. 252–259, 262, 263, 265, 840, 888
 - \list@clearing@reg 261
 - \list@create 237–242,
244–246, 685, 716, 717, 1379, 1512
 - \list@pstartL@pc .. 716, 719, 729, 730
 - \list@pstartR@pc .. 716, 724, 735, 736
 - \listbreak 2234, 2238
 - \listxadd 360, 2508–2511
 - \lock@disp 1238, 1242, 1247
 - \lock@off 497, 498, 506, 648, 649
 - \lock@on 644, 645
 - \luatexbodydir 829, 877, 1033
 - \luatexpardir 828, 876, 1032
 - \luatextextdir .. 827, 875, 1015, 1031
- M**
- \managestanza@modulo 1700
 - \maxchunks 3, 1724
 - \maxlinesinpar@list 244, 265
 - \memorydump 8, 778, 795
 - \memorydumpL 120, 778
 - \memorydumpR 120, 795
 - \message 55
 - \multiply 1212, 1999, 2048
- N**
- \n@num 534, 653
 - \n@num@reg 540

- \namebox 977, 982, 1063,
1068, 1708, 1828, 1831, 2424, 2461
 - \NeedsTeXFormat 2
 - \new@line 1041, 1044
 - \new@lineL 606, 1017
 - \new@lineR 608
 - \newbool 683, 684
 - \newbox 811, 961, 962, 1709
 - \newcommandx .. 819, 867, 914, 936, 1688
 - \newcounter 93–
96, 181, 183, 185, 187, 814, 816
 - \newhookcommand@series 707, 709
 - \newif 6, 33, 137,
138, 590, 770, 771, 959, 960,
1367, 1630, 1765, 1812, 1824,
1957, 1958, 2319, 2321, 2420, 2421
 - \newlength 1985, 1988
 - \newmarks 2512–2514
 - \newnamebox 1708, 1736, 1737
 - \newnamecount 1719, 1744
 - \newsavebox 1848, 1849
 - \newseries@eledpar 680, 712
 - \newwrite 589, 2501
 - \next@absline
.... 1960, 1962, 1964, 2371–2373
 - \next@abslineR
.... 1961, 1963, 1965, 2383–2385
 - \next@action 282
 - \next@actionline 279, 281
 - \next@actionlineR
. 271, 273, 1137, 1175, 1197, 1199
 - \next@actionR 274, 1138,
1176, 1177, 1182, 1183, 1191, 1200
 - \next@insert 841
 - \next@insertR
889, 1383, 1386, 1388, 1391, 1395
 - \next@page@num 359, 431
 - \next@page@numR .. 53, 295, 297, 349, 428
 - \noindent 865, 912, 934, 956
 - \normal@page@break 360
 - \normal@page@breakR 46
 - \normal@pars 70, 844, 892
 - \normalbfnoteX 1603
 - \notefontsetup 2558
 - \noteschanged@true
..... 75, 78, 662, 671, 1385
 - \notesXwidthliketwocolumns 4
 - \num@lines 922, 1858, 2087
 - \num@linesR 810, 944, 1859, 2088
 - \numberedpar@true 860, 907
 - \numberingRfalse 69
 - \numberingRtrue 43, 104, 130
 - \numberingtrue 122
 - \numberpstartfalse 9
 - \numberpstarttrue 9
 - \numdef 691, 1026, 1573, 1580,
1960, 1961, 2371, 2383, 2571, 2573
 - \numgdef 1545, 1552, 2392, 2403
 - \numlabfont 226
 - \numpagelinesL 2069,
2136, 2159, 2163, 2328, 2394, 2591
 - \numpagelinesR
2069, 2177, 2200, 2204, 2349, 2405
- O**
- \old@footnote 699, 703
 - \old@otherlanguage 1802
 - \old@startstanza .. 780, 781, 803, 804
 - \oldchapter 753, 763
 - \one@line
. 982, 985, 1012, 1019, 1041, 1044
 - \one@lineR 810, 1068, 1071
 - \onlysideX 6
 - \onlyXside 6
 - \openout 64, 597, 603
 - \otherlanguage 1802, 1803
- P**
- \p@pstartL 862
 - \p@pstartR 909
 - \PackageError 19
 - \page@action 296, 425, 554
 - \page@num 277, 357, 1325
 - \page@numR 248, 269, 347,
560, 565, 1177, 1300, 1346, 1568
 - \pagebreak 1974
 - \pagegoal 2418
 - \Pages 5, 2073
 - pages (environment) 5, 740
 - \pagetotal 2141, 2143, 2182,
2184, 2250, 2256, 2265, 2327, 2348
 - pairs (environment) 3, 740
 - \paperwidth 1040, 1043
 - \par@line 923, 1860, 2089
 - \par@lineR 810, 945, 1861, 2090
 - \parbox 1882, 1893
 - \parledgroup@ .. 984, 1070, 2512, 2538
 - \parledgroup@beforenotes@save ..
..... 931, 953, 2598
 - \parledgroup@beforenotesL 2596

- \parledgroup@beforenotesR [2596](#)
 - \parledgroup@correction@notespacing [2146](#), [2187](#), [2585](#)
 - \parledgroup@correction@notespacing@final [2454](#), [2489](#), [2566](#)
 - \parledgroup@correction@notespacing@initial [2164](#), [2205](#), [2561](#), [2569](#)
 - \parledgroup@notes@endL [2427](#), [2430](#), [2453](#), [2531](#)
 - \parledgroup@notes@endR [2464](#), [2467](#), [2488](#), [2534](#)
 - \parledgroup@notes@startL [984](#), [2515](#), [2531](#)
 - \parledgroup@notes@startR [1070](#), [2515](#), [2531](#)
 - \parledgroup@notespacing@correction [2556](#), [2588](#)–[2590](#)
 - \parledgroup@notespacing@correction@accumulated [2562](#), [2568](#), [2589](#)
 - \parledgroup@notespacing@correction@modulo [2563](#), [2590](#)–[2592](#)
 - \parledgroup@notespacing@set@correction [2076](#), [2556](#)
 - \parledgroup@series [2513](#), [2517](#), [2518](#), [2525](#), [2526](#), [2541](#), [2542](#), [2548](#), [2549](#)
 - \parledgroup@type [2514](#), [2517](#), [2518](#), [2525](#), [2526](#), [2539](#), [2546](#)
 - \parledgroupnotespacing [2555](#), [2558](#)
 - \parledgroupseries@ [2512](#)
 - \parledgrouptrue [11](#)
 - \parledgrouptype@ [2512](#)
 - \pausenumbering [793](#)
 - \pausenumberingR [103](#), [793](#)
 - \pend [7](#), [777](#), [798](#), [837](#), [1690](#)
 - \pendL [777](#), [914](#)
 - \pendR [798](#), [885](#), [936](#)
 - \prev@abslineverse [2392](#)–[2394](#), [2403](#)–[2405](#)
 - \prev@nopbR [2502](#)
 - \prev@pbR [2502](#)
 - \prevgraf [922](#), [944](#), [1858](#), [1859](#), [2087](#), [2088](#)
 - \print@columnseparator [1901](#), [1937](#)
 - \print@eledsectionL [1022](#), [1882](#), [2134](#)
 - \print@eledsectionR [1096](#), [1893](#), [2175](#)
 - \print@last@after@pendLfalse [2340](#)
 - \print@last@after@pendLtrue [2456](#)
 - \print@last@after@pendRfalse [2361](#)
 - \print@last@after@pendRtrue [2491](#)
 - \print@lineL [997](#), [1007](#)
 - \print@lineR [1083](#), [1096](#)
 - \print@notesX [2096](#)
 - \print@notesX@forpages [1425](#), [2096](#)
 - \print@Xnotes [2095](#)
 - \print@Xnotes@forpages [1425](#), [2095](#)
 - \printlines [1423](#)
 - \printlinesR [9](#), [1412](#)
 - \ProcessOptions [13](#)
 - \protected@csxdef [701](#)
 - \protected@edef [861](#), [908](#)
 - \protected@write [1771](#)
 - \protected@xdef [1612](#), [1623](#)
 - \ProvidesPackage [3](#)
 - \pst@rtedLfalse [32](#)
 - \pst@rtedLtrue [123](#), [842](#)
 - \pst@rtedRfalse [34](#), [42](#), [72](#)
 - \pst@rtedRtrue [107](#), [131](#), [890](#)
 - \pstart [7](#), [21](#), [25](#), [775](#), [797](#), [1684](#)
 - \pstartL [775](#), [813](#)
 - \pstartnumfalse [1360](#), [1365](#)
 - \pstartnumRfalse [1372](#), [1377](#)
 - \pstartnumRtrue [1368](#), [1867](#), [2486](#)
 - \pstartnumtrue [1866](#), [2451](#)
 - \pstartR [797](#), [813](#)
- R**
- \Rcolwidth [4](#), [5](#), [14](#), [759](#), [906](#), [1079](#), [1089](#), [1893](#), [1997](#), [2018](#), [2035](#), [2046](#), [2062](#), [2083](#)
 - \read@linelist [250](#), [593](#)
 - \rem@inder [1282](#), [1284](#)–[1286](#)
 - \RequirePackage [5](#)
 - \resetprevline@ [1918](#), [1922](#), [2440](#), [2477](#)
 - \restore@pstartL@pc [728](#), [1870](#), [2117](#), [2452](#)
 - \restore@pstartR@pc [728](#), [1871](#), [2118](#), [2487](#)
 - \resumenumbering [794](#)
 - \resumenumberingR [103](#), [794](#)
 - \rightlinenumR [219](#), [1295](#), [1303](#)
 - \rightpstartnumL [1313](#)
 - \rightpstartnumR [1313](#)
 - Rightside (environment) [7](#), [789](#)
 - \Rightsidehook [784](#), [802](#)
 - \Rightsidehookend [784](#), [807](#)
 - \rlap [1295](#), [1303](#), [1320](#), [1328](#), [1341](#), [1349](#)
 - \Rlineflag [9](#), [214](#), [226](#), [1415](#), [1419](#), [1518](#)
 - \rule [1977](#)

S

- \savebox 1882, 1893
- \sc@n@list 1283, 1285
- \secdef 768
- \section@num 124–126
- \section@numR 30,
44, 55, 56, 63, 64, 108–110, 132–134
- \select@language 1769, 1771–1774, 1808
- \selectlanguage 1777
- \set@line 659
- \set@line@action
..... 289, 394, 403, 410, 433, 556
- \setl@dlp@rbox 1561, 1576, 1578
- \setl@drp@rbox 1563, 1571, 1583
- \setline 627
- \setlinenum 635
- \setnamebox 850, 898, 1708
- \setnotepositionliketwocolumns@C
..... 1991
- \setnotepositionliketwocolumns@L
..... 1991
- \setnotepositionliketwocolumns@R
..... 1991
- \setnotespositionliketwocolumns@C
..... 2030
- \setnotespositionliketwocolumns@L
..... 2008
- \setnotespositionliketwocolumns@R
..... 2057
- \setpositionliketwocolumns@C ...
..... 1991, 2025
- \setpositionliketwocolumns@L ...
..... 1991, 2004
- \setpositionliketwocolumns@R ...
..... 1991, 2053
- \setprintlines 1413
- \setwidthliketwocolumns@C 1991, 2012
- \setwidthliketwocolumns@L 1991, 1991
- \setwidthliketwocolumns@R 1991, 2042
- \shiftedpstartsfalse 8
- \shiftedpstartstrue 7, 9, 10
- \shiftedversesfalse 8
- \shiftedversestrue 7
- \sidenote@margin 1538, 1541
- \sidenote@marginR 1535, 1566
- \sidenotecontent@
..... 1544, 1549, 1550, 1561, 1563,
1571, 1572, 1576, 1578, 1579, 1583
- \sidenotemargin 1535
- \sidenotemargin* 1535
- \sidenotesep 1550
- \skip 1441, 1449, 1485, 1493, 2541, 2548
- \skip@lockoff 498, 506
- \skipnumbering 10, 652
- \skipnumbering@reg 656
- \smash 1977
- \splitbotmarks 2538,
2539, 2541, 2542, 2546, 2548, 2549
- \splitfirstmarks
984, 1070, 2517, 2518, 2525, 2526
- \splittopskip .. 979, 1065, 1461, 1504
- \stanza@count 1668, 1681, 1695
- \stanza@hang 1670, 1703
- \stanza@modulo 1668, 1698
- \stanzaindentbase
..... 1648, 1658, 1696, 1699
- \startlock 643
- \startstanzahook 1666
- \startsub 610
- \sub@action 305, 454, 555
- \sub@change 54, 299, 300, 306
- \sub@lock 1120
- \sub@lockR 51, 314, 316, 318,
321, 472, 478, 479, 481, 482,
512, 513, 515, 1102, 1158, 1160,
1161, 1163, 1219, 1259, 1261, 1263
- \sub@off 618, 619
- \sub@on 612, 613
- \subline@num 228, 355, 373,
374, 376, 406, 446, 1121, 1126, 1531
- \subline@numR 229,
233, 322, 326, 345, 367, 368,
370, 399, 437, 562, 566, 1103,
1108, 1179, 1186, 1273, 1274, 1527
- \sublinenumincrement 7, 190
- \sublinenumincrement* 7, 190
- \sublinenumr@p . 1416, 1420, 1527, 1531
- \sublinenumrepR 216, 229
- \sublines@false 52, 303, 1148
- \sublines@true 301, 1146
- \sublock@disp 1221, 1225, 1230
- \sw@list@inedtextR 242, 259
- \sw@listR 241, 258
- \symplinenum 1415
- \sza@penalty 1676, 1680

T

- \temp 1994, 1995, 1998–2001,
2014, 2015, 2020–2022, 2031,

2033, 2036–2039, 2043, 2044, 2047–2050, 2058, 2060, 2063–2066	\vbadness 978, 1064
\temp@ 1026, 1027	\vbfnoteX 1613, 1624
\temp@spacing 2558, 2559	\vbox . . . 850, 898, 1462, 1505, 1882, 1893
\tempa 2032, 2034–2036, 2059, 2061–2063	\vl@dbfnote 1593, 1597
\textwidth 15, 17, 758, 759, 2082, 2083	\vsplit 982, 1068, 1463, 1506
\theledlanguageL . . . 1777, 1874, 2127	
\theledlanguageR . . . 1777, 1885, 2167	W
\thepage 607, 609	\wd 1012
\thepstart 776, 796	\widthliketwocolumns 4
\thepstartL	\widthliketwocolumnstrue 12
. 9, 776, 815, 855, 862, 1359, 1364	\WithSuffix 165, 210–213, 1535
\thepstartR	\writtenlinesLfalse 2093, 2437
. 9, 796, 817, 902, 909, 1371, 1376	\writtenlinesLtrue 2434
\thisfootnote . . 1612, 1613, 1623, 1624	\writtenlinesRfalse 2094, 2474
\thr@@ . . 481, 490, 513, 520, 1153, 1161	\writtenlinesRtrue 2471
\togglefalse 689	
\toggletrue 687	X
\topskip 2250	\xifinlist 995, 1027, 1081, 1876, 1887, 2133, 2174
	\xifinlistcs 1962– 1965, 1968–1971, 2367, 2368, 2372, 2373, 2379, 2380, 2384, 2385
U	\Xnoteswidthliketwocolumns 4
\unhbox 1713, 1897, 1902, 2139, 2147, 2180, 2188	\xpg@main@language 1809, 1810
\unhnamebox 1708	\xright@appenditem 427, 428, 430, 431, 435, 442, 444, 451, 456, 458, 460, 463, 465, 467, 475, 477, 486, 509, 511, 518, 537, 538, 548, 564, 576, 580, 584, 587, 692, 719, 724, 1526, 1530, 1593, 1597, 1613, 1624
\unvbox . . . 985, 1071, 1465, 1508, 1715	\xspace 704
\unvnamebox 1708	
\usebox 1899, 1904	
\usenamecount 1669, 1675, 1719, 1751, 2104, 2426, 2429, 2446, 2448, 2463, 2466, 2481, 2483, 2516, 2524	
V	
\value 1694	

Change History

v0.1.	v0.3.
General: First public release 1	General: Added \do@lineLhook and \do@lineRhook 46
v0.2.	Reorganize for ledarab 1
General: Added section of babel re- lated code 66	\affixline@numR: Changed \affixline@numR to match new eledmac 49
Fix babel problems 1	\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR 48
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns 70	\do@lineL: Added \do@lineLhook
\Pages: Added \l@duselanguage to \Pages 77	

to <code>\do@lineL</code>	44	v0.3.c.	
Simplified <code>\do@lineL</code> by using macros for some common code	44	General: Compatibilty with Poly- glossia	1
<code>\do@lineR</code> : Changed <code>\do@lineR</code> similarly to <code>\do@lineL</code>	46	v0.4.	
Leftside: Added hooks into Left- side environment	38	General: No more ledparpatch. All patches are now in the main file.	1
<code>\flag@end</code> : Removed extraneous spaces from <code>\flag@end</code>	32	v0.5.	
<code>\ifledRcol</code> : Moved <code>\ifl@pairing</code> to <code>eledmac</code>	15	General: Corrections about <code>\section</code> and other titles in numbered sections	1
<code>\ifpst@rtedR</code> : Moved <code>\ifpst@rtedL</code> to <code>eledmac</code>	16	v0.6.	
<code>\l@dlinenumR</code> : Simplified <code>\leftlinenumR</code> and <code>\rightlinenumR</code> by introducing <code>\l@dlinenumR</code>	21	General: Be able to use <code>\chapter</code> in parallel pages.	1
<code>\l@dnumstartsR</code> : Moved <code>\l@dnumstartsL</code> to <code>eledmac</code> .	65	v0.7.	
<code>\ledsavedprintlines</code> : Simpli- fied <code>\printlinesR</code> by using <code>\setprintlines</code>	55	General: Option ‘shiftedverses’ which make there is no blank between two parallel verses with inequal length.	1
<code>\ledstrutR</code> : Added <code>\ledstrutL</code> and <code>\ledstrutR</code>	80	v0.8.	
<code>\normalbfnoteX</code> : Removed extraneous spaces from <code>\normalbfnoteX</code>	62	General: Possibility to have a sym- bol on each hanging of verses, like in the french typogra- phy. Redefine the commande <code>\hangingsymbol</code> to define the character.	1
<code>\Pages</code> : Added <code>\ledstrutL</code> to <code>\Pages</code>	77	v0.9.	
Added <code>\ledstrutR</code> to <code>\Pages</code> .	78	General: Possibility to number <code>\pstart</code>	9
<code>\Rightsidehookend</code> : Added <code>\Leftsidehook</code> , <code>\Leftsidehookend</code> , <code>\Rightsidehook</code> and <code>\Rightsidehookend</code>	38	Possibilty to number the <code>pstart</code> with the commands <code>\numberpstarttrue</code>	1
<code>\sublinenumrepR</code> : Added <code>\linenumrepR</code> and <code>\sublinenumrepR</code>	21	<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code>	15
v0.3.a.		v0.9.1.	
General: Minor <code>\linenummargin</code> fix	1	General: The numbering of the <code>pstarts</code> restarts on each <code>\beginnumbering</code>	1
<code>\line@marginR</code> : Don’t just set <code>\line@marginR</code> in <code>\linenummargin</code>	19	v0.9.2.	
v0.3.b.		General: Debug : with <code>\Columns</code> , the hanging indentation now runs on the left columns and the hanging symbol is shown only when <code>\stanza</code> is used.	1
General: Improved parallel page balancing	1	v0.9.3.	
<code>\Pages</code> : Added <code>\l@dminpagelines</code> calculation for succeeding page pairs	79	General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes con- flicts with memoir class.	1

v0.10.		v1.1.1.	
General: <code>\edlabel</code> commands on the right side are now correctly indicated.	1	<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1.	40
<code>\edlabel</code> commands which start a paragraph are now put in the right place.	1	v1.1.2.	
v0.11.		<code>\affixside@noteR</code> : Remove spurious space between line number and line content	60
General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by <code>pstart</code> (like in <code>eledmac 0.15</code>).	44	v1.2.	
Lineation can be by <code>pstart</code> (like in <code>eledmac 0.15</code>).	18	General: Support for <code>\led<section></code> commands in parallel texts. . . .	1
New management of hangingsymbol insertion, preventing undesirable insertions.	62	v1.2.1.	
Prevent shift of column separator when a verse is hanged	63	<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once.	17
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code>).	49	v1.3.	
<code>\Columns</code> : Line numbering by <code>pstart</code>	71	<code>\edtext</code> : Manage RTL language. . .	34
<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code>).	85	v1.3.1.	
<code>Pstart</code> number can be printed in side	86	<code>\l@dbfnote</code> : Compatibility of standard footnotes with <code>eledmac</code> when theses footnotes contain any commands.	61
v0.12.		v1.3.2.	
General: New new management of hangingsymbol insertion, preventing undesirable insertions. . . .	62	General: Debug with some classes. .	1
v1.0.		v1.3.3.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code> . . .	1	General: Debugging the left notes of the right column.	60
Debug in lineation by <code>pstart</code> . .	18	<code>\l@dbfnote</code> : Spurious space with footnote in right column. . . .	61
v1.0.1.		v1.3.4.	
General: Correction on <code>\numberonlyfirstinline</code> with lineation by <code>pstart</code> or by page.	1	General: Allow use of commands in sidenotes, as introduced by <code>eledmac 1.0</code>	60
v1.1.		v1.3.5.	
General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code>	1	<code>\normalbfnoteX</code> : Allows one to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded.	62
<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from <code>eledmac</code>).	40	v1.4.	
		General: Added <code>\do@insidelineLhook</code> and <code>\do@insidelineRhook</code> . . .	46
		v1.4.1.	
		<code>\normalbfnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text. . .	62
		<code>astanza</code> : Enable the use of <code>stanzaindent</code> repetition within <code>astanza</code> environment.	63
		v1.4.3.	
		General: Corrects a false hanging verse when a verse is exactly the	

length of a line.	1	New sectioning commands, as in	
<code>\inserthangingsymbolR</code> : Hang		eledmac.	13
verse is now not automatically		<code>\Columns</code> : Modify <code>\Columns</code> to en-	
flush right.	62	able to add section's title. . . .	69
<code>\pendL</code> : Spurious spaces in <code>\pendL</code> .	42	Suppress <code>\l@dchecklang</code> from	
<code>\pendR</code> : Spurious spaces in		<code>\Columns</code>	70
<code>\pstartR</code>	43	<code>\l@dchecklang</code> : Suppress	
<code>\pstartR</code> : Spurious spaces in		<code>\l@dchecklang</code> which didn't	
<code>\pstartL</code> and <code>\pstartR</code>	40	work and was not logical, be-	
v1.5.0.		cause both columns could have	
General: Add, as in eledmac, fea-		the same language but not the	
tures to manage page breaks. . .	1	main language of the docu-	
<code>\sublinenumincrement*</code> : Add		ment.	67
starred version of <code>\firstlinenum</code> ,		<code>\Pages</code> : Modify <code>\Pages</code> to enable to	
<code>\linenumincrement</code> , <code>\firstsublinenum</code> ,		add section's title.	75
<code>\sublinenumincrement</code> to		<code>\pendL</code> : As in eledmac, <code>\pendL</code> can	
change both Left and Right-		have an optional argument. . .	42
side.	20	<code>\pendR</code> : As in eledmac, <code>\pendR</code> can	
v1.6.0.		have an optional argument. . .	43
General: Add tool and documenta-		<code>\print@columnseparator</code> : Move	
tion for parallel ledgroups . . .	12	some code of <code>\Columns</code> to	
v1.7.0.		<code>\print@columnseparator</code> . . .	72
General: Add, as in eledmac, fea-		<code>\pstartR</code> : As in eledmac, <code>\pendL</code>	
tures to make crossrefs with		and <code>\pendR</code> can have an op-	
<code>pstart</code> numbers.	1	tional argument.	40
v1.8.0.		<code>\sidenotemargin*</code> : <code>\sidenotemargin</code>	
General: <code>\beginnumbering</code> is de-		is now directly defined in eled-	
fined only on eledmac, not on		mac to be able to manage eled-	
eledpar.	16	par.	60
<code>\l@dlsnote</code> , <code>\l@drsnote</code> and		Add <code>\sidenotemargin*</code>	60
<code>\l@dcnote</code> defined only one		<code>\theledlanguageR</code> : Correct	
time, in eledmac.	60	left/right language setting with	
Add <code>\beforecolumnseparator</code>		polyglossia.	68
and <code>\aftercolumnseparator</code> . . .	4	v1.8.1.	
Add <code>\columnspan</code>	4	<code>\do@lineL</code> : Fix a bug with critical	
Add, as in eledmac, new system		notes at the beginning of a page,	
of sectioning commands.	1	(maybe added by v1.8.0) (?). .	44
Add, as in eledmac, option to in-		<code>\do@lineR</code> : Fix a bug with critical	
sert something after <code>\pends</code> /		notes at the beginning of a page,	
verses.	1	added by v1.8.0 (?).	46
Add, as in eledmac, option		v1.8.2.	
to insert something between		General: Debug <code>\eledxxx</code> with	
<code>\pstarts</code> / verse.	1	some paper sizes	1
Change <code>\do@lineR</code> and		Debug left and side note (bugs	
<code>\do@lineR</code> to allow new sec-		added by 1.8.0)	1
tioning commands.	44	<code>\eledpar@error</code> : Errors specific to	
Compatibility with musixtex. . .	1	eledpar send to eledpar hand-	
Debug eledmac section-		book	15
ing command after using		<code>\flag@end</code> : <code>\flag@start</code> and	
<code>\resumenumbering</code>	1	<code>\flag@end</code> are now defined only	

one time for eledmac and eled-	vertical mode to solve spurious
par	space before minipage.
\lineation*: Add \lineation*	32
v1.8.3.	v1.11.0.
General: Add \noeledxxx, as in	General: Compatibility of standard
eledmac	footnotes with some biblatex
\doinsidelineRhook: Added	styles.
\dolineLhook, \dolineRhook,	\edtext: \critext and \edtext are
\doinsidelineLhook and	now defined only in eledmac.
\doinsidelineRhook	34
\Pages: Debug blank pages when	v1.12.0.
using optional argument in the	General: Compatibility with
last \pend.	Lua \LaTeX RTL languages.
75	\Columns: Add \l@drprintingcolumnstrue
\resumenumberingR: Debug
\resumenumberingR	69
17	\edlabel: \edlabel and \edindex
v1.9.0.	works now with hyperref when
General: Add \AtBeginPairs	using eledpar.
macro.	59
Compatibility with \Xnoteswidthliketwocolumns	\edlabel is now defined only one
and \notesXwidthliketwocolumns	time for both eledmac and eled-
.	par
1	59
\ifwidthliketwocolumns: Added	\Pages: Add \l@drprintingpagetrue
widthliketwocolumns option
14	75
\theledlanguageR: Debug	\print@eledsectionL: Compati-
left/right language switching	bility with Lua \LaTeX RTL lan-
with polyglossia. Don't write in	guages.
.aux file when setting left/right	45
lines.	\print@eledsectionR: Compati-
68	bility with Lua \LaTeX RTL lan-
v1.9.1.	guages.
\ifledRcol: Moved \ifl@drpacing	47
to eledmac	\print@lineL: Compatibility with
15	Lua \LaTeX RTL languages.
v1.10.0.	45
General: Compatibility with	v1.12.1.
\AtEveryPstart and \AtEveryPend	\print@eledsectionL: Fixes
.	bug with Lua \LaTeX RTL
1	\eledsection.
Restore critical notes in	45
\eledsection in parallel	v1.13.0.
columns (this bug was added	General: Fix bug in shiftedpstarts
in 1.8.2).	when size difference between
1	pstarts is very important.
\Pages: Debug wrong pages split-	1
ting when no optional argument	With parallel pages, long notes
is used in last \pend (bug was	can now flow from the Left
added in v.1.8.3).	to the right side and from the
75	Right to the left side.
Debug wrong parallel pages syn-	1
chronization when an \edtext	\clearl@drighthouse: Use
falls accross two pages.	\newpage instead of
75	\clearpage.
v1.10.1.	81
\line@list@stuffR: Revert modi-	\ifledRcol: Remove false boolean
fication of 1.4.2, which makes	settings which are not needed.
bugs with numbering. Leave	15
	\Pages: Prevent false overfull
	hboxes when using \Pages out-
	side of pages environment.
	76

When using shiftedpstarts option, a <code>\l@ddleftbox</code> with a null height will advance the <code>\pagetotal</code> in any case.	75	<code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> directly in <code>\print@notesX@forpages</code> and <code>\print@Xnotes@forpages</code> . 56
astanza: Enable the use of optional argument of <code>&</code> in <code>astanza</code> environment.	63	Correct <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> . 56
v1.13.1. <code>\correct@footinsX@box:</code> Call		<code>\Pages:</code> Prevent false empty page after <code>\Pages</code> (bug added in 1.13.0) 75