

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

Maïeul Rouquette[†]based on the original **ledpar** by Peter Wilson
Herries Press[‡]

This is documentation of deprecated eledmac package. If you are beginning a new project, we suggest that you use reledmac instead. If for old projects you can't migrate to reledmac, you can continue to use this documentation and the eledmac package. You should add noreledmac option when loading package, to disable message about reledmac.

Abstract

The **eledmac** package, which is based on the PLAIN \TeX 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). Examples starting by “3-” are for basic uses, those starting by “4-” are for advanced uses.

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

*This file (**eledpar.dtx**) has version number v1.17.1, last revised 2015/09/01.

[†]maieul at maieul dot net

[‡]herries dot press at earthlink dot net

2 The <code>eledpar</code> package	5
2.1 General	5
3 Parallel columns	6
4 Facing pages	7
4.1 Basic usage	7
4.2 Text width	8
4.3 Page number	8
4.4 Setting the page breaking	8
4.5 Critical and familiar footnotes	8
4.5.1 Note size setting	9
4.5.2 Notes for one side only	9
4.5.3 Familiar notes called in the right side, but to be printed in the left side	9
5 Left and right texts	10
5.1 Environments	10
5.2 Line numbering scheme	10
5.3 chunk	10
5.4 <code>\AtEveryPstart</code> and <code>\AtEveryPstartCall</code>	11
5.5 Language setting	11
5.6 Shifting	11
6 Numbering text lines and paragraphs	12
7 Verse	13
8 Side notes	15
9 Parallel ledgroups	15
9.1 Parallel ledgroups and <code>setspace</code> package	17
10 Sectioning commands	17
11 Implementation overview	17
12 Preliminaries	17
12.1 Messages	19
13 Sectioning commands	19
14 Line counting	22
14.1 Choosing the system of lineation	22
14.2 Line-number counters and lists	26
14.2.1 Correspond to those in <code>eledmac</code> for regular or left text . . .	26
14.2.2 Specific to <code>eledpar</code>	26
14.3 Reading the line-list file	26

<i>Contents</i>	3
14.4 Commands within the line-list file	28
14.5 Writing to the line-list file	36
15 Marking text for notes	38
15.1 Specific hooks and commands for notes	39
15.1.1 Notes to be printed on one side only	39
15.1.2 Familiar footnotes without marks	39
15.1.3 Create hooks	40
15.1.4 Init standards series (A,B,C,D,E,Z)	41
16 Pstart numbers dumping and restoration	41
17 Parallel environments	42
18 Paragraph decomposition and reassembly	44
18.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	44
18.2 Processing one line	48
18.3 Line and page number computation	52
18.4 Line number printing	55
18.5 Pstart number printing in side	57
18.6 Add insertions to the vertical list	58
18.7 Penalties	59
18.8 Printing leftover notes	60
19 Footnotes	60
19.1 Normal footnote formatting	60
19.2 Footnotes output specific to <code>\Pages</code>	61
20 Cross referencing	65
21 Side notes	65
22 Familiar footnotes	67
23 Verse	68
24 Naming macros	70
25 Counts and boxes for parallel texts	70
26 Fixing babel	71
27 Parallel columns	74

28 Parallel pages	81
28.1 Specific counters	81
28.2 Main macro	82
28.3 Ensure all notes be printed at the end of parallel pages	86
28.4 Struts	87
28.5 Page clearing	87
28.6 Lines managing	88
28.7 Page break managing	89
28.8 Getting boxes content	92
28.9 Same page number in both side	94
29 Page break/no page break, depending on the specific line	95
30 Parallel ledgroup	96
31 The End	99
Appendix A Some things to do when changing version	100
Appendix A.1 Migration to <code>eledpar</code> 1.4.3	100
References	100
Index	100
Change History	120

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 31); 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.

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

Keep in mind that the `\Columns` **must be** outside of the `pairs` environment.

`\AtBeginPairs` 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.
`\columnseparator` 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`
`\aftercolumnseparator`

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

- 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 revert to the previous behavior, just set with a negative value. If you want to mix two-column with single-column text, you can align horizontally single-column text to two-column text with `\widthliketwocolumnstrue`. To reset this feature, use `\widthliketwocolumnsfalse`. You can also call `\widthliketwocolumns` as a global option when loading `eledmac` or `eledpar`.

`\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` 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 Page number

By default, `\Pages` use the standard L^AT_EX page number scheme. This means that pages are numbered continuously following printed-book conventions: from left-hand to right-hand side, left-hand pages having even numbers, right-hand pages having odd numbers.

`\sameparallelpagenumbertrue` However, you can use the package option `sameparallelpagenumber` to have the same page number for both left and right side. In this case, this setting will apply only for pages typeset by `\Pages`, not for “normal” pages.

You can also switch the two system using `\sameparallelpagenumbertrue` and `\sameparallelpagenumberfalse`.

4.4 Setting the page breaking

`\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}
```

4.5 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.

4.5.1 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 (5.4.9 p. 32). As an advisable setting:

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

4.5.2 Notes for one side only

`\onlyXside` You may want to typeset notes on one side only (either left or right). Use `\onlyXside[⟨s⟩]{⟨p⟩}` to set critical notes, and `\onlysideX[⟨s⟩]{⟨p⟩}` to set familiar notes. `{⟨p⟩}` 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.

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

`\footnoteXnomk` 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.

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:

```
\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 ...
```

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


```

\pend
\endnumbering
\end{Rightside}

```

5 Left and right texts

5.1 Environments

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 likewise in the **Rightside** environment. The number of **Leftside** and **Rightside** environments must be the same.

5.2 Line numbering scheme

```

\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement
\firstlinenum*
\linenumincrement*
\firstsublinenum*
\sublinenumincrement*

```

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.

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.

5.3 chunk

```

\pstart
\pend

```

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.

5.4 `\AtEveryPstart` and `\AtEveryPstartCall`

In general, remember that the moment where a `\pstart` is called is different from the moment when the `\pstart... \pend` content is printed, which is when `\Pages` or `\Columns` is processed.

Consequently:

- The argument of `\AtEveryPstart` (see 4.2.3 p. 14) is called before every chunk is printed, except if you used an optional argument for the `\pstart`.
- The argument of `\AtEveryPstartCall` is called before every `\pstart`.

5.5 Language setting

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.

5.6 Shifting

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


<code>\Rlineflag</code>	<p>The value of <code>\Rlineflag</code> is appended to the line numbers of the right texts. Its default definition is:</p> <pre>\newcommand*{\Rlineflag}{R}</pre> <p>This may be useful for parallel columns but for parallel pages it might be more appropriate to redefine it as:</p>
<code>\printlinesR</code>	<p><code>\renewcommand*{\Rlineflag}{}</code>. The <code>\printlines</code> 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 <code>\printlinesR</code>, which incorporates <code>\Rlineflag</code>. (The macro <code>\ledsavedprintlines</code> is a copy of the original <code>\printlines</code>, just in case ...). As provided, the package makes no use of <code>\printlinesR</code> 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 <code>\Rlineflag</code>. You could do this by putting the following code in your preamble:</p> <pre>\let\oldBfootfmt\Bfootfmt \renewcommand{\Bfootfmt}[3]{% \let\printlines\printlinesR \oldBfootfmt{#1}{#2}{#3}}</pre>
<code>\ledsavedprintlines</code>	
<code>\numberpstarttrue</code>	<p>It's possible to insert a number at every <code>\pstart</code> command. You must use the <code>\numberpstarttrue</code> command to have it. You can stop the numerotation with <code>\numberpstartfalse</code>. You can redefine the commands <code>\thepstartL</code> and <code>\thepstartR</code> to change style. The numbering restarts on each <code>\beginnumbering</code></p>
<code>\numberpstartfalse</code>	
<code>\thepstartL</code>	
<code>\thepstartR</code>	

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 `\sza000`’ it is because you have forgotten to use `\setstanzaindent` 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 be 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]{%
  \hskip -#1\llap{\textbf{#2}}\hskip #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}
\end{pages}
\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

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).

By default, the L^AT_EX marks for header are taken 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 LaTeX2_ε. The package also requires the *eledmac* package.


```

1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{eledpar}[2015/09/01 v1.17.1 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}

```

The `parledgroup` can be called either on `eledmac` or `eledpar`.

```
11 \DeclareOption{parledgroup}{\parledgrouptrue}
```

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

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

`\ifsameparallelpagenumber`

```

13 \newif\ifsameparallelpagenumber%
14 \DeclareOption{sameparallelpagenumber}{\sameparallelpagenumbertrue}%

```

```
15 \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 we 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` is also set TRUE if we are doing parallel pages. `\ifledRcol` is set TRUE if we are doing the right hand text. They are defined in `eledmac`.

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

```

\lcolwidth
\lcolwidth
16 \newdimen\Lcolwidth
17 \Lcolwidth=0.45\textwidth
18 \newdimen\Rcolwidth
19 \Rcolwidth=0.45\textwidth
20

```


12.1 Messages

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

```

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

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

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

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

\led@err@Leftside@PreviousNotPrinted
\led@err@Rightside@PreviousNotPrinted
32 \newcommand*{\led@err@Leftside@PreviousNotPrinted}{%
33   \eledpar@error{You call a new Leftside environment while the previous one has not been typeset by \string\pstart}{\@ehc}}
34 \newcommand*{\led@err@Rightside@PreviousNotPrinted}{%
35   \eledpar@error{You call a new Rightside environment while the previous one has not been typeset by \string\pstart}{\@ehc}}

\led@err@Pages@InsideEnv
\led@err@Columns@InsideEnv
36 \newcommand*{\led@err@Pages@InsideEnv}{%
37   \eledpar@error{\string\Pages\space must be called *outside* of the `pages` environment}{\@ehc}}
38 \newcommand*{\led@err@Columns@InsideEnv}{%
39   \eledpar@error{\string\Columns\space must be called *outside* of the `pairs` environment}{\@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.

```

40 \newcount\section@numR
41 \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`.

```
42 \pst@rtedLfalse
43 \newif\ifpst@rtedR
44
```

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

```
45 \newcommand*{\beginnumberingR}{%
46   \ifnumberingR
47     \led@err@NumberingStarted
48     \endnumberingR
49   \fi
50   \global\l@dnumpststartsR \z@
51   \global\pst@rtedRfalse
52   \global\numberingRtrue
53   \global\advance\section@numR \@ne
54   \global\absline@numR \z@
55   \gdef\normal@page@breakR{}
56   \gdef\l@prev@pbR{}
57   \gdef\l@prev@nopbR{}
58   \global\line@numR \z@
59   \global\@lockR \z@
60   \global\sub@lockR \z@
61   \global\sublines@false
62   \global\let\next@page@numR\relax
63   \global\let\sub@change\relax
64   \message{Section \the\section@numR R }%
65   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
66   \l@dend@stuff
67   \setcounter{pstartR}{1}
68   \begingroup
69   \initnumbering@sectcountR
70   \gdef\eled@sectionsR@@{}%
71   \if@noeled@sec\else%
72     \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@numR R}{-}{-}\makeatother%
73     \immediate\openout\eled@sectioningR@out=\jobname.eledsec\the\section@numR R\relax%
74   \fi%
75 }
```

`\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.

```
76 \def\endnumberingR{%
77   \ifnumberingR
```



```

78 \global\numberingRfalse
79 \normal@pars
80 \ifnum\l@dnumstartR=0%
81 \led@err@NumberingWithoutPstart%
82 \fi%
83 \ifl@dpairing
84 \global\pst@rtedRfalse
85 \else
86 \ifx\insertlines@listR\empty\else
87 \global\noteschanged@true
88 \fi
89 \ifx\line@listR\empty\else
90 \global\noteschanged@true
91 \fi
92 \fi
93 \ifnoteschanged@
94 \led@mess@NotesChanged
95 \fi
96 \else
97 \led@err@NumberingNotStarted
98 \fi
99 \endgroup
100 \if@noeled@sec\else%
101 \immediate\closeout\eled@sectioningR@out%
102 \fi%
103 }
104

```

`\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 L^AT_EX counter in `\numberingR`.

```

105 \newcounter{chapterR}
106 \newcounter{sectionR}
107 \newcounter{subsectionR}
108 \newcounter{subsubsectionR}
109 \newcommand{\initnumbering@sectcountR}{
110 \let\c@chapter\c@chapterR
111 \let\c@section\c@sectionR
112 \let\c@subsection\c@subsectionR
113 \let\c@subsubsection\c@subsubsectionR
114 }

```

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

```

115 \newcommand*{\pausenumberingR}{%
116 \endnumberingR\global\numberingRtrue}
117 \newcommand*{\resumenumberingR}{%
118 \ifnumberingR
119 \global\pst@rtedRtrue
120 \global\advance\section@numR \@ne

```



```

121 \led@mess@SectionContinued{\the\section@numR R}%
122 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
123 \l@dend@stuff
124 \begin@group%
125 \initnumbering@sectcountR%
126 \else
127 \led@err@numberingShouldHaveStarted
128 \endnumberingR
129 \beginnumberingR
130 \fi}
131

```

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

```

132 \newcommand*{\memorydumpL}{%
133 \endnumbering
134 \numberingtrue
135 \global\pst@rtedLtrue
136 \global\advance\section@num \@ne
137 \led@mess@SectionContinued{\the\section@num}%
138 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
139 \l@dend@stuff}
140 \newcommand*{\memorydumpR}{%
141 \endnumberingR
142 \numberingRtrue
143 \global\pst@rtedRtrue
144 \global\advance\section@numR \@ne
145 \led@mess@SectionContinued{\the\section@numR R}%
146 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
147 \l@dend@stuff}
148

```

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.

<code>\ifbypstart@R</code>	The <code>\ifbypage@R</code> and <code>\ifbypstart@R</code> flag specify the current lineation system:
<code>\bypstart@Rtrue</code>	
<code>\bypstart@Rfalse</code>	• line-of-page : <code>bypstart@R = false</code> and <code>bypage@R = true</code> .
<code>\ifbypage@R</code>	
<code>\bypage@Rtrue</code>	• line-of-pstart : <code>bypstart@R = true</code> and <code>bypage@R = false</code> .
<code>\bypage@Rfalse</code>	<code>eledpar</code> will use the line-of-section system unless instructed otherwise.


```

149 \newif\ifbypage@R
150 \newif\ifbypstart@R

```

`\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`.

```

151 \newcommand*{\lineationR}[1]{%
152   \ifnumbering
153     \led@err@LineationInNumbered
154   \else
155     \def\@tempa{#1}\def\@tempb{page}%
156     \ifx\@tempa\@tempb
157       \global\bypage@Rtrue
158       \global\bypstart@Rfalse
159       \unless\ifnocritical@%
160         \pstartinfootnote[] [false]%
161       \fi%
162     \else
163       \def\@tempb{pstart}%
164       \ifx\@tempa\@tempb
165         \global\bypage@Rfalse
166         \global\bypstart@Rtrue
167         \unless\ifnocritical@%
168           \pstartinfootnote%
169         \fi%
170       \else
171         \def\@tempb{section}
172         \ifx\@tempa\@tempb
173           \global\bypage@Rfalse%
174           \global\bypstart@Rfalse%
175           \unless\ifnocritical@%
176             \pstartinfootnote[] [false]%
177           \fi%
178         \else
179           \led@warn@BadLineation
180         \fi%
181       \fi
182     \fi
183   \fi}}

```

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

```

184 \WithSuffix\newcommand\lineation*[1]{%
185   \lineation{#1}%
186   \lineationR{#1}%
187 }%

```

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

`\line@marginR`

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.

```

188 \newcount\line@marginR
189 \renewcommand*{\linenummargin}[1]{%
190   \l@getline@margin{#1}%
191   \ifnum\l@dttempcntb>\m@ne
192     \ifledRcol
193       \global\line@marginR=\l@dttempcntb
194     \else
195       \global\line@margin=\l@dttempcntb
196     \fi
197   \fi}}

```

By default put right text numbers at the right.

```

198 \line@marginR=\@ne
199

```

`\c@firstlinenumR` The following counters tell `eledmac` which right text lines should be printed with 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.

```

200 \newcounter{firstlinenumR}
201 \setcounter{firstlinenumR}{5}
202 \newcounter{linenumincrementR}
203 \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.

```

204 \newcounter{firstsublinenumR}
205 \setcounter{firstsublinenumR}{5}
206 \newcounter{sublinenumincrementR}
207 \setcounter{sublinenumincrementR}{5}
208

```

`\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` 209 `\providecommand*{\firstlinenum}{}`
`\firstlinenum*` 210 `\providecommand*{\linenumincrement}{}`
`\linenumincrement*` 211 `\providecommand*{\firstsublinenum}{}`
`\firstsublinenum*` 212 `\providecommand*{\sublinenumincrement}{}`
`\sublinenumincrement*` 213 `\renewcommand*{\firstlinenum}[1]{%`
214 `\ifledRcol \setcounter{firstlinenumR}{#1}%`
215 `\else \setcounter{firstlinenumR}{#1}%`
216 `\fi}`


```

217 \renewcommand*{\linenumincrement}[1]{%
218   \ifledRcol \setcounter{linenumincrementR}{#1}%
219   \else      \setcounter{linenumincrement}{#1}%
220   \fi}
221 \renewcommand*{\firstsublinenum}[1]{%
222   \ifledRcol \setcounter{firstsublinenumR}{#1}%
223   \else      \setcounter{firstsublinenum}{#1}%
224   \fi}
225 \renewcommand*{\sublinenumincrement}[1]{%
226   \ifledRcol \setcounter{sublinenumincrementR}{#1}%
227   \else      \setcounter{sublinenumincrement}{#1}%
228   \fi}
229 \WithSuffix\newcommand\firstlinenum*[1]{\setcounter{firstlinenumR}{#1}\setcounter{firstlinenum}{#1}}
230 \WithSuffix\newcommand\linenumincrement*[1]{\setcounter{linenumincrementR}{#1}\setcounter{linenumincr
231 \WithSuffix\newcommand\firstsublinenum*[1]{\setcounter{subfirstlinenumR}{#1}\setcounter{subfirstlinenum
232 \WithSuffix\newcommand\sublinenumincrement*[1]{\setcounter{sublinenumincrementR}{#1}\setcounter{subli

```

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

```

233 \newcommand*{\Rlineflag}{R}
234

```

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

```

235 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
236 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
237

```

`\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`.

```

238 \newcommand*{\leftlinenumR}{%
239   \l@dlinenumR
240   \kern\linenumsep}
241 \newcommand*{\rightlinenumR}{%
242   \kern\linenumsep
243   \l@dlinenumR}
244 \newcommand*{\l@dlinenumR}{%
245   \numlabfont\linenumrepR{\line@numR}\Rlineflag%
246   \ifsublines@
247     \ifnum\subline@num>\z@
248       \unskip\fullstop\sublinenumrepR{\subline@numR}%
249     \fi
250   \fi}
251

```


14.2 Line-number counters and lists

14.2.1 Correspond to those in `eledmac` for regular or left text

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.

```
252 \newcount\line@numR
253 \newcount\subline@numR
254 \newcount\absline@numR
255
```

`\line@listR` `\insertlines@listR` `\actionlines@listR` Now we can define the list macros that will be created from the line-list file. They are directly analagous 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:

```
256 \list@create{\line@listR}
257 \list@create{\insertlines@listR}
258 \list@create{\actionlines@listR}
259 \list@create{\actions@listR}
260
```

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

```
261 \newcount\page@numR
262
```

14.2.2 Specific to `eledpar`

`\linesinpar@listL` `\linesinpar@listR` `\maxlinesinpar@list` 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.

```
263 \list@create{\linesinpar@listL}
264 \list@create{\linesinpar@listR}
265 \list@create{\maxlinesinpar@list}
266
```

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.

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


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

```

268 \ifledRcol
269 \list@clear{\line@listR}%
270 \list@clear{\insertlines@listR}%
271 \list@clear{\actionlines@listR}%
272 \list@clear{\actions@listR}%
273 \list@clear{\linesinpar@listR}%
274 \list@clear{\linesonpage@listR}
275 \else
276 \list@clearing@reg
277 \list@clear{\linesinpar@listL}%
278 \list@clear{\linesonpage@listL}%
279 \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).

```

280 \list@clear{\maxlinesinpar@list}

```

Now get the file and interpret it.

```

281 \get@linelistfile{#1}%
282 \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.

```

283 \ifledRcol
284 \global\page@numR=\m@ne
285 \ifx\actionlines@listR\empty
286 \gdef\next@actionlineR{1000000}%
287 \else
288 \gl@p\actionlines@listR\to\next@actionlineR
289 \gl@p\actions@listR\to\next@actionR
290 \fi
291 \else
292 \global\page@num=\m@ne
293 \ifx\actionlines@list\empty
294 \gdef\next@actionline{1000000}%
295 \else
296 \gl@p\actionlines@list\to\next@actionline
297 \gl@p\actions@list\to\next@action
298 \fi
299 \fi}
300

```

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.

```

301 \newcommand{\@nl@regR}{%
302   \ifx\l@dchset@num\relax \else
303     \advance\absline@numR \@ne
304     \set@line@action
305     \let\l@dchset@num\relax
306     \advance\absline@numR \m@ne
307     \advance\line@numR \m@ne%    % do we need this?
308   \fi
309   \advance\absline@numR \@ne
310   \ifx\next@page@numR\relax \else
311     \page@action
312     \let\next@page@numR\relax
313   \fi
314   \ifx\sub@change\relax \else
315     \ifnum\sub@change>\z@
316       \sublines@true
317     \else
318       \sublines@false
319     \fi
320     \sub@action
321     \let\sub@change\relax
322   \fi
323   \ifcase\@lockR
324   \or
325     \@lockR \tw@
326   \or\or
327     \@lockR \z@
328   \fi
329   \ifcase\sub@lockR
330   \or
331     \sub@lockR \tw@
332   \or\or
333     \sub@lockR \z@
334   \fi
335   \ifsublines@
336     \ifnum\sub@lockR<\tw@
337       \advance\subline@numR \@ne
338     \fi
339   \else
340     \ifnum\@lockR<\tw@

```



```

341     \advance\line@numR \@ne \subline@numR \z@
342     \fi
343 \fi}
344
345 \renewcommand*{\@nl}[2]{%
346   \fix@page{#1}%
347   \ifledRcol
348     \@nl@regR
349   \else
350     \@nl@reg
351   \fi}
352

```

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

```

\fix@page
353 \newcount\last@page@numR
354 \last@page@numR=-10000
355 \renewcommand*{\fix@page}[1]{%
356   \ifledRcol
357     \ifnum #1=\last@page@numR
358     \else
359       \ifbypage@R
360         \line@numR \z@ \subline@numR \z@
361       \fi
362       \page@numR=#1\relax
363       \last@page@numR=#1\relax
364       \def\next@page@numR{#1}%
365     \fi
366   \else
367     \ifnum #1=\last@page@num
368     \else
369       \ifbypage@
370         \line@num \z@ \subline@num \z@
371       \fi
372       \page@num=#1\relax
373       \last@page@num=#1\relax
374       \def\next@page@num{#1}%
375       \listxadd{\normal@page@break}{\the\absline@num}
376     \fi
377   \fi}
378

```

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

```

379 \renewcommand*{\@adv}[1]{%
380   \ifsublines@
381     \ifledRcol
382       \advance\subline@numR by #1\relax
383       \ifnum\subline@numR<\z@
384         \led@warn@BadAdvancelineSubline

```



```

385     \subline@numR \z@
386     \fi
387   \else
388     \advance\subline@num by #1\relax
389     \ifnum\subline@num<\z@
390       \led@warn@BadAdvancelineSubline
391       \subline@num \z@
392     \fi
393   \fi
394 \else
395   \ifledRcol
396     \advance\line@numR by #1\relax
397     \ifnum\line@numR<\z@
398       \led@warn@BadAdvancelineLine
399       \line@numR \z@
400     \fi
401   \else
402     \advance\line@num by #1\relax
403     \ifnum\line@num<\z@
404       \led@warn@BadAdvancelineLine
405       \line@num \z@
406     \fi
407   \fi
408 \fi
409 \set@line@action}
410

```

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

```

411 \renewcommand*{\@set}[1]{%
412   \ifledRcol
413     \ifsublines@
414       \subline@numR=#1\relax
415     \else
416       \line@numR=#1\relax
417     \fi
418     \set@line@action
419   \else
420     \ifsublines@
421       \subline@num=#1\relax
422     \else
423       \line@num=#1\relax
424     \fi
425     \set@line@action
426   \fi}
427

```

`\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.

```

428 \renewcommand*{\l@d@set}[1]{%
429   \ifledRcol
430     \line@numR=#1\relax
431     \advance\line@numR \@ne
432     \def\l@dchset@num{#1}
433   \else
434     \line@num=#1\relax
435     \advance\line@num \@ne
436     \def\l@dchset@num{#1}
437   \fi}
438 \let\l@dchset@num\relax
439
```

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

```

440 \renewcommand*{\page@action}{%
441   \ifledRcol
442     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
443     \xright@appenditem{\next@page@numR}\to\actions@listR
444   \else
445     \xright@appenditem{\the\absline@num}\to\actionlines@list
446     \xright@appenditem{\next@page@num}\to\actions@list
447   \fi}

```

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

```

448 \renewcommand*{\set@line@action}{%
449   \ifledRcol
450     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
451     \ifsublines@
452       \@l@dttempcnta=-\subline@numR
453     \else
454       \@l@dttempcnta=-\line@numR
455     \fi
456     \advance\@l@dttempcnta by -5000\relax
457     \xright@appenditem{\the\@l@dttempcnta}\to\actions@listR
458   \else
459     \xright@appenditem{\the\absline@num}\to\actionlines@list
460     \ifsublines@
461       \@l@dttempcnta=-\subline@num
462     \else
463       \@l@dttempcnta=-\line@num
464     \fi
465     \advance\@l@dttempcnta by -5000\relax
466     \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
467   \fi}
468
```


`\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.

```

469 \renewcommand*{\sub@action}{%
470   \ifledRcol
471     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
472     \ifsublines@
473       \xright@appenditem{-1001}\to\actions@listR
474     \else
475       \xright@appenditem{-1002}\to\actions@listR
476     \fi
477   \else
478     \xright@appenditem{\the\absline@num}\to\actionlines@list
479     \ifsublines@
480       \xright@appenditem{-1001}\to\actions@list
481     \else
482       \xright@appenditem{-1002}\to\actions@list
483     \fi
484   \fi}
485
```

`\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.

```

486 \newcount\@lockR
487 \newcount\sub@lockR
488
489 \newcommand*{\do@lockonR}{%
490   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
491   \ifsublines@
492     \xright@appenditem{-1005}\to\actions@listR
493     \ifnum\sub@lockR=\z@
494       \sub@lockR \@ne
495     \else
496       \ifnum\sub@lockR=\thr@@
497         \sub@lockR \@ne
498       \fi
499     \fi
500   \else
501     \xright@appenditem{-1003}\to\actions@listR
502     \ifnum\@lockR=\z@
503       \@lockR \@ne
504     \else
505       \ifnum\@lockR=\thr@@
506         \@lockR \@ne
507       \fi
508     \fi
509   \fi}
510
511 \renewcommand*{\do@lockon}{%

```



```

512 \ifx\next\lock@off
513   \global\let\lock@off=\skip@lockoff
514 \else
515   \ifledRcol
516     \do@lockonR
517   \else
518     \do@lockonL
519   \fi
520 \fi}

```

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

```

\do@lockoff 521
\do@lockoffR 522
\skip@lockoff 523 \newcommand{\do@lockoffR}{%
524   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
525   \ifsublines@
526     \xright@appenditem{-1006}\to\actions@listR
527     \ifnum\sub@lockR=\tw@
528       \sub@lockR \thr@@
529     \else
530       \sub@lockR \z@
531     \fi
532   \else
533     \xright@appenditem{-1004}\to\actions@listR
534     \ifnum\@lockR=\tw@
535       \@lockR \thr@@
536     \else
537       \@lockR \z@
538     \fi
539   \fi}
540
541 \renewcommand*{\do@lockoff}{%
542   \ifledRcol
543     \do@lockoffR
544   \else
545     \do@lockoffL
546   \fi}
547 \global\let\lock@off=\do@lockoff
548

```

`\n@num`

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

`\insert@countR` 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`.

```

549   \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.

```

550 \renewcommand*{\@ref}[2]{%
551   \ifledRcol
552     \global\advance\@edtext@level by 1%
553     \global\insert@countR=#1\relax
554     \loop\ifnum\insert@countR>\z@
555       \xright@appenditem{\the\absline@numR}\to\insertlines@listR
556       \global\advance\insert@countR \m@ne
557     \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.

```

558 \begingroup
559   \let\@ref=\dummy@ref
560   \let\@lopR\@gobble
561   \let\page@action=\relax
562   \let\sub@action=\relax
563   \let\set@line@action=\relax
564   \let\@lab=\relax
565   \let\@lemma=\relax
566   \let\@sw\@gobblethree%
567   #2
568   \global\endpage@num=\page@numR
569   \global\endline@num=\line@numR
570   \global\endsubline@num=\subline@numR
571 \endgroup

```

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

```

572 \xright@appenditem%
573   {\the\page@numR|\the\line@numR|}%
574   \ifsublines@ \the\subline@numR \else 0\fi|}%
575   \the\endpage@num|\the\endline@num|}%
576   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR

```

Create a list which will store all the second argument of each `\@sw` in this lemma, at this level.

```

577 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\@edtext@level\end

```

Declare and init boolean for lemma in this level.

```

578 \providebool{lemmacommand@\the\@edtext@level}%
579 \boolfalse{lemmacommand@\the\@edtext@level}%

```


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

```

580      #2
581 % Now, we store the list of \cs{@sw} of this current \cs{edtext} as an element of
582 % the global list of list of \cs{@sw} for a \cs{edtext} depth.
583 %   \begin{macrocode}
584   \ifnum\@edtext@level>0%
585   \def\create@this@edtext@level{\expandafter\list@create\expandafter{\csname sw@list@edtextR@the\@
586   \ifcsundef{sw@list@edtextR@the\@edtext@level}{\create@this@edtext@level}{}}%
587   \letcs{\@tmp}{sw@list@edtextR@the\@edtext@level}%
588   \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}%
589   \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
590   \global\cslet{sw@list@edtextR@the\@edtext@level}{\@tmp}%
591   \fi%

```

Decrease edtext level counter.

```

592   \global\advance\@edtext@level by -1%
593 \else
    And when not in right text
594   \@ref@reg{#1}{#2}%
595 \fi}

```

`\@pend` `\@pend{<num>}` adds its argument to the `\linesinpar@listL` list, and analogously 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.

```

596 \providecommand*\@pend{[1]}{}
597 \renewcommand*\@pend{[1]}{%
598   \ifbypstart@\global\line@num=0\fi%
599   \xright@appenditem{#1}\to\linesinpar@listL}
600 \providecommand*\@pendR{[1]}{}
601 \renewcommand*\@pendR{[1]}{%
602   \ifbypstart@R\global\line@numR=0\fi
603   \xright@appenditem{#1}\to\linesinpar@listR}
604

```

`\@lopL` `\@lopL{<num>}` adds its argument to the `\linesonpage@listL` list, and analogously 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.

```

605 \providecommand*\@lopL{[1]}{}
606 \renewcommand*\@lopL{[1]}{%
607   \xright@appenditem{#1}\to\linesonpage@listL}
608 \providecommand*\@lopR{[1]}{}
609 \renewcommand*\@lopR{[1]}{%
610   \xright@appenditem{#1}\to\linesonpage@listR}
611

```


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`.

```
612 \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
```

```
613 \newif\iffirst@linenum@out@R
```

```
614 \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.

```
615 \newcommand*{\line@list@stuffR}[1]{%
```

```
616 \read@linelist{#1}%
```

```
617 \iffirst@linenum@out@R
```

```
618 \immediate\closeout\linenum@outR
```

```
619 \global\first@linenum@out@Rfalse
```

```
620 \immediate\openout\linenum@outR=#1
```

```
621 \immediate\write\linenum@outR{\string\line@list@version{\this@line@list@version}}%
```

```
622 \else
```

```
623 \if@minipage%
```

```
624 \leavevmode%
```

```
625 \fi%
```

```
626 \closeout\linenum@outR%
```

```
627 \openout\linenum@outR=#1%
```

```
628 \fi}
```

```
629
```

`\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.

```
630 \newcommand*{\new@lineL}{%
```

```
631 \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.

```
632 \newcommand*{\new@lineR}{%
```

```
633 \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 send the `\@ref` command to the line-list file.

```
\flag@end
```

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

```
\endsub
```

```
634 \renewcommand*{\startsub}{\dimen0\lastskip
```



```

635 \ifdim\dimen0>0pt \unskip \fi
636 \ifledRcol \write\linenum@outR{\string\sub@on}%
637 \else      \write\linenum@out{\string\sub@on}%
638 \fi
639 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
640 \def\endsub{\dimen0\lastskip
641 \ifdim\dimen0>0pt \unskip \fi
642 \ifledRcol \write\linenum@outR{\string\sub@off}%
643 \else      \write\linenum@out{\string\sub@off}%
644 \fi
645 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
646

```

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

```

647 \renewcommand*{\advanceline}[1]{%
648 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
649 \else      \write\linenum@out{\string\@adv[#1]}%
650 \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.

```

651 \renewcommand*{\setline}[1]{%
652 \ifnum#1<\z@
653 \led@warn@BadSetline
654 \else
655 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
656 \else      \write\linenum@out{\string\@set[#1]}%
657 \fi
658 \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.

```

659 \renewcommand*{\setlinenum}[1]{%
660 \ifnum#1<\z@
661 \led@warn@BadSetlinenum
662 \else
663 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
664 \else      \write\linenum@out{\string\l@d@set[#1]} \fi
665 \fi}
666

```

\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.

```

667 \renewcommand*{\startlock}{%
668 \ifledRcol \write\linenum@outR{\string\lock@on}%
669 \else      \write\linenum@out{\string\lock@on}%
670 \fi}

```



```

671 \def\endlock{%
672   \ifledRcol \write\linenum@outR{\string\lock@off}%
673   \else      \write\linenum@out{\string\lock@off}%
674   \fi}
675

```

\skipnumbering

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.

676 \renewcommand*{\set@line}{%
677   \ifledRcol
678     \ifx\line@listR\empty
679       \global\noteschanged@true
680       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
681     \else
682       \gl@p\line@listR\to\@tempb
683       \xdef\l@d@nums{\@tempb|\edfont@info}%
684       \global\let\@tempb=\undefined
685     \fi
686   \else
687     \ifx\line@list\empty
688       \global\noteschanged@true
689       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
690     \else
691       \gl@p\line@list\to\@tempb
692       \xdef\l@d@nums{\@tempb|\edfont@info}%
693       \global\let\@tempb=\undefined
694     \fi
695   \fi}
696

```


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`

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

15.1.1 Notes to be printed on one side only

`eledpar` allows notes to be printed on one side only. We need to declare these options. We also need boolean flags, and to set them to true when a note series is not printed on one side. We check the `nofamiliar` and `nocritical` `eledmac` options.

```
698 \unless\ifnofamiliar%
699 \csgdef{onlysideX@#1}{}%
700 \global\newbool{keepforsideX@#1}%
701 \fi%
702 \unless\ifnocritical%
703 \global\newbool{keepforXside@#1}%
704 \csgdef{onlyXside@#1}{}%
705 \fi%
```

15.1.2 Familiar footnotes without marks

The `\footnoteXnomk` commands are for notes which are printed on 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.

First, check the `nofamiliar` option of `eledmac`

```
706 \unless\ifnofamiliar%
707 % So declare the list.
708 % \begin{macrocode}
709 \expandafter\list@create\csname footnote#1@mk\endcsname%
```

Then, declare the `\footnoteXnomk` command.

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

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

```
711 \toggletrue{nomk}%
712 \csuse{footnote#1}{##1}%
713 \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`.


```

714      \letcs{\@tmp}{footnote#1@mk}%
715      \numdef\@tmpa{\csuse{c@footnote#1}}%
716      \global\xright@appenditem{\@tmpa}\to\@tmp%
717      \global\cslet{footnote#1@mk}{\@tmp}%
718  }%

```

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

```

719      \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.

```

720      \letcs{\@tmp}{footnote#1@mk}%
721      \gl@p\@tmp\to\@tmpa%
722      \global\cslet{footnote#1@mk}{\@tmp}%

```

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

```

723      \letcs{\old@footnote}{c@footnote#1}%
724      \setcounter{footnote#1}{\@tmpa}%

```

Define the footnote mark and print it

```

725      \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
726      \csuse{@footnotemark#1}%

```

Restore previous footnote counter and finally add space.

```

727      \setcounter{footnote#1}{\old@footnote}%
728      \xspace%
729  }%

```

End of tools for familiar notes without marks

```

730  \fi

```

End of `\newseries@eledpar`.

```

731 }%

```

15.1.3 Create hooks

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

```

732 \unless\ifnocritical%
733   \newhookcommand@series{onlyXside}%
734 \fi%
735 \unless\ifnofamiliar%
736   \newhookcommand@series{onlysideX}%
737 \fi
738
739

```


15.1.4 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`.

```

740 \newcommand{\init@series@eledpar}{%
741   \def\do##1{\newseries@eledpar{##1}}%
742   \dolistloop{\@series}%
743 }%
744 \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 745 \list@create{\list@pstartL@pc}%
746 \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 747 \def\dump@pstartL@pc{%
748   \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
749   \global\cslet{numberpstart@L\the\l@dumpstartsL}{\ifnumberpstart}%
750 }%
751
752 \def\dump@pstartR@pc{%
753   \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
754   \global\cslet{numberpstart@R\the\l@dumpstartsR}{\ifnumberpstart}%
755 }%
756
```

`\restore@pstartL@pc` And so, the commands to restore them

```

\restore@pstartR@pc 757 \def\restore@pstartL@pc{%
758   \ifx\list@pstartL@pc\empty\else%
759     \gl@p\list@pstartL@pc\to\@temp%
760     \global\c@pstartL=\@temp%
761   \fi%
762 }%
```



```

763 \def\restore@pstartR@pc{%
764   \ifx\list@pstartR@pc\empty\else%
765     \gl@p\list@pstartR@pc\to\@temp%
766     \global\c@pstartR=\@temp%
767   \fi%
768 }%

```

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
pages    parallel pages.
chapterinpages 769 \newenvironment{pairs}{%}
              770   \l@dpairingtrue
              771   \l@dpagingfalse
              772   \initnumbering@sectcmd
              773   \at@begin@pairs%
              774 }{%
              775   \l@dpairingfalse
              776 }
              777

\AtBeginPairs The \AtBeginPairs macro just define a \at@begin@pairs macro, called at the
              beginning of each pairs environments.
              778 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
              779 \def\at@begin@pairs{}%
              780

```

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.

```

781 \newenvironment{pages}{%
782   \let\oldchapter\chapter
783   \let\chapter\chapterinpages
784   \l@dpairingtrue
785   \l@dpagingtrue
786   \initnumbering@sectcmd
787   \setlength{\Lcolwidth}{\textwidth}%
788   \setlength{\Rcolwidth}{\textwidth}%
789 }{%
790   \l@dpairingfalse
791   \l@dpagingfalse
792   \let\chapter\oldchapter
793 }
794 \newcommand{\chapterinpages}{\thispagestyle{plain}%
795                               \global\@topnum\z@

```



```

796 \afterindentfalse
797 \secdef\@chapter\@schapter}
798

```

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

```

799 \newif\ifinstanzaL
800 \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.

```

801 \newenvironment{Leftside}{%
802 \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
803 \led@err@Leftside@PreviousNotPrinted%
804 \fi%
805 \ledRcolfalse
806 \setcounter{pstartL}{1}
807 \let\pstart\pstartL
808 \let\thepstart\thepstartL
809 \let\pend\pendL
810 \let\memorydump\memorydumpL
811 \Leftsidehook
812 \let\old@startstanza\@startstanza
813 \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
814 }{
815 \Leftsidehookend}

```

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

```

\Leftsidehookend
\Rightsidehook 816 \newcommand*{\Leftsidehook}{}
\Rightsidehookend 817 \newcommand*{\Leftsidehookend}{}
818 \newcommand*{\Rightsidehook}{}
819 \newcommand*{\Rightsidehookend}{}
820

```

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.

```

821 \newenvironment{Rightside}{%
822 \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
823 \led@err@Rightside@PreviousNotPrinted%
824 \fi%
825 \ledRcoltrue
826 \let\beginnumbering\beginnumberingR
827 \let\endnumbering\endnumberingR
828 \let\pausenumbering\pausenumberingR
829 \let\resumenumbering\resumenumberingR

```



```

830 \let\memorydump\memorydumpR
831 \let\thepstart\thepstartR
832 \let\pstart\pstartR
833 \let\pend\pendR
834 \let\ledpb\ledpbR
835 \let\lednopb\lednopbR
836 \let\lineation\lineationR
837 \Rightsidehook
838 \let\old@startstanza\@startstanza
839 \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
840 }f%
841 \ledRcolfalse
842 \Rightsidehookend
843 }
844

```

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.

```
845 \newcount\num@linesR
```

```
846 \newbox\one@lineR
```

```
847 \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.

```

848
849 \newcounter{pstartL}
850 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
851 \newcounter{pstartR}
852 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
853
854 \newcommandx*{\pstartL}[1][1]{%
855   \if@nobreak%
856     \let\@oldnobreak\@nobreaktrue%
857   \else%
858     \let\@oldnobreak\@nobreakfalse%
859   \fi%
860   \@nobreaktrue%
861   \ifluatex%
862     \xdef\l@luatextextdir@L{\the\textdir}%
863     \xdef\l@luatexpardir@L{\the\pardir}%
864     \xdef\l@luatexbodydir@L{\the\bodydir}%
865   \fi%
866   \ifnumbering \else%
867     \led@err@PstartNotNumbered%
868     \beginnumbering%
869   \fi%
870   \ifnumberedpar%
871     \led@err@PstartInPstart%
872   \pend%
873 \fi%
```

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

```

874 \ifpst@rtedL\else%
875   \list@clear{\inserts@list}%
876   \global\let\next@insert=\empty%
877   \global\pst@rtedLtrue%
878 \fi%
879 \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.

```

880 \global\advance\l@dnumpstartsL \@ne%
881 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
882   \led@err@TooManyPstarts%
883   \global\l@dnumpstartsL=\l@dc@maxchunks%
884 \fi%
885 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup%
```

We set all the usual interline penalties to zero; 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.


```

886 \l@dzero penalties%
887 \ifautopar\else%
888   \ifnumberpstart%
889     \ifsidepstartnum%
890     \else%
891       \thepstartL%
892     \fi%
893   \fi%
894 \fi%
895 \hsize=\Lcolwidth%
896 \numberedpar@true%
897 \iflabelpstart\protected@edef\@currentlabel%
898   {\p@pstartL\thepstartL}\fi%
  Dump the optional arguments
899 \ifstrempy{#1}%
900   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\at@every@pstart}}%
901   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent#1}}%
902   \at@every@pstart@call%
903 }
904 \newcommand*{\pstartR}[1][1]{%
905   \if@nbreak%
906     \let\@oldnbreak\@nbreaktrue%
907   \else%
908     \let\@oldnbreak\@nbreakfalse%
909   \fi%
910   \@nbreaktrue%
911   \ifluatex%
912     \xdef\l@luatextextdir@R{\the\textdir}%
913     \xdef\l@luatexpardir@R{\the\pardir}%
914     \xdef\l@luatexbodydir@R{\the\bodydir}%
915   \fi%
916   \ifnumberingR \else%
917     \led@err@PstartNotNumbered%
918     \beginnumberingR%
919   \fi%
920   \ifnumberedpar@%
921     \led@err@PstartInPstart%
922     \pendR%
923   \fi%
924   \ifpst@rtedR\else%
925     \list@clear{\inserts@listR}%
926     \global\let\next@insertR=\empty%
927     \global\pst@rtedRtrue%
928   \fi%
929   \begingroup\normal@pars%
930   \global\advance\l@dnumpstartsR \@ne%
931   \ifnum\l@dnumpstartsR>\l@dc@maxchunks%
932     \led@err@TooManyPstarts%
933     \global\l@dnumpstartsR=\l@dc@maxchunks%

```



```

934 \fi%
935 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
936 \l@dzeropenalties%
937 \ifautopar\else%
938 \ifnumberpstart%
939 \ifsidepstartnum\else%
940 \thepstartR%
941 \fi%
942 \fi%
943 \fi%
944 \hsize=\Rcolwidth%
945 \numberedpar@true%
946 \iflabelpstart\protected@edef\@currentlabel%
947 {\p@pstartR\thepstartR}\fi%
948 \ifstrempy{#1}%
949 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
950 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
951 \at@every@pstart@call%
952 }

```

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

```

953 \newcommand*{\pendL}[1][1]{%
954 \ifnumbering \else%
955 \led@err@PendNotNumbered%
956 \fi%
957 \ifnumberedpar@ \else%
958 \led@err@PendNoPstart%
959 \fi%

```

We 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.

```

960 \endgraf\global\num@lines=\prevgraf\egroup%
961 \global\par@line=0%

```

End the group that was begun in the `\pstart`.

```

962 \endgroup%
963 \ignorespaces%
964 \@oldnobreak%
965 \dump@pstartL@pc%
966 \ifnumberpstart%
967 \addtocounter{pstartL}{1}%
968 \fi
969 \parledgroup@beforenotes@save{L}%

```

Dump content of the optional argument.

```

970 \ifstrempy{#1}%
971 {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
972 {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
973 }

```


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

```

974 \newcommandx*{\pendR}[1][1]{%
975   \ifnumberingR \else%
976     \led@err@PendNotNumbered%
977   \fi%
978   \ifnumberedpar@ \else%
979     \led@err@PendNoPstart%
980   \fi%
981   \endgraf\global\num@linesR=\prevgraf\egroup%
982   \global\par@lineR=0%
983   \endgroup%
984   \ignorespaces%
985   \@oldnobreak%
986   \dump@pstartR@pc%
987   \ifnumberpstart%
988     \addtocounter{pstartR}{1}%
989   \fi%
990   \parledgroup@beforenotes@save{R}%
991   \ifstrempy{#1}%
992     {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
993     {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
994 }
995
```

`\AtEveryPstartCall` The `\AtEveryPstartCall` argument is called when the `\pstartL` or `\pstartR` is called. That is different of `\AtEveryPstart` the argument of which is called when the `\pstarts` are printed.

```

996 \newcommand{\AtEveryPstartCall}[1]{\xdef\at@every@pstart@call{\unexpanded{#1}}}%
997 \gdef\at@every@pstart@call{}%
```

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

```

998 \newif\ifprint@last@after@pendL%
999 \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 analagously for a line of right text.

```

1000 \newbox\l@dleftbox
1001 \newbox\l@drightbox
1002
```

`\countLline` We need to know the number of lines processed.

```

\countRline 1003 \newcount\countLline
```



```

1004 \countLline \z@
1005 \newcount\countRline
1006 \countRline \z@
1007

```

\@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 1008 \newcount\@donereallinesL
1009 \newcount\@donetotallinesL
1010 \newcount\@donereallinesR
1011 \newcount\@donetotallinesR
1012

```

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

```

1013 \newcommand*{\do@lineL}{%
1014 \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
1015 \advance\countLline \@ne%
1016 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
1017 {\vbadness=10000%
1018 \splittopskip=\z@%
1019 \do@lineLhook%
1020 \l@emptyd@ta%
1021 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}%
1022 to\baselineskip}%
1023 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startL}{}%
1024 \unvbox\one@line \global\setbox\one@line=\lastbox%
1025 \getline@numL%
1026 \ifnum\@lock>\@ne%
1027 \inserthangingsymboltrue%
1028 \else%
1029 \inserthangingsymbolfalse%
1030 \fi%
1031 \setbox\l@dleftbox%
1032 \hb@xt@ \Lcolwidth{%
1033 \ifl@dhidenumber%
1034 \global\l@dhidenumberfalse%
1035 \f@x@l@cks%
1036 \else%
1037 \affixline@num%
1038 \fi%
1039 \xifinlist{\the\l@dpscl}{\eled@sections@}%
1040 {\add@inserts\affixside@note}%
1041 {\print@lineL}}%
1042 \add@penaltiesL%
1043 \global\advance\@donereallinesL\@ne%
1044 \global\advance\@donetotallinesL\@ne%
1045 \else%

```



```

1046 \setbox\l@leftbox \hb@xt@ \lcolwidth{\hspace*\lcolwidth}}%
1047 \global\advance\@donetotallinesL\@ne%
1048 \fi}
1049
1050

```

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

```

1051 \def\print@lineL{%
1052   \affixstart@numL%
1053   \l@dld@ta %space kept for backward compatibility
1054   \add@inserts\affixside@note%
1055   \l@dlsn@te %space kept for backward compatibility
1056   {\ledllfill\hb@xt@ \lcolwidth{%
1057     \do@insidelineLhook%
1058     \ifluatex%
1059       \textdir\l@luatexttexdir@L%
1060     \fi%
1061     \new@lineL%
1062     \inserthangingsymbolL%
1063     \l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%
1064   \l@drsn@te}}
1065

```

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

```

1066 \def\print@eledsectionL{%
1067   \addtocounter{pstartL}{-1}%
1068   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
1069   \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}%
1070   \numdef{\temp@}{\l@dpscL-1}%
1071   \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1072   \@eled@sectioningtrue%
1073   \bgroup%
1074   \ifluatex%
1075     \textdir\l@luatexttexdir@L%
1076     \pardir\l@luatexpardir@L%
1077     \bodydir\l@luatexbodydir@L%
1078     \ifdefstring{\l@luatexttexdir@L}{TRT}{\@RTLtrue}{%
1079   \fi%
1080   \csuse{eled@sectioning@\the\l@dpscL}%
1081   \egroup%
1082   \@eled@sectioningfalse%
1083   \global\csundef{eled@sectioning@\the\l@dpscL}%
1084   \if@RTL%
1085     \hspace{-3\paperwidth}%
1086     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1087   \else%
1088     \hspace{3\paperwidth}%
1089     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1090   \fi%

```



```

1091 \vskip\eledsection@correcting@skip%
1092 }
1093

```

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

```

\doininsidelineLhook 1094 \newcommand*\dolineLhook[1]{\gdef\dolineLhook{#1}}%
\doininsidelineRhook 1095 \newcommand*\dolineRhook[1]{\gdef\dolineRhook{#1}}%
1096 \newcommand*\doininsidelineLhook[1]{\gdef\doininsidelineLhook{#1}}%
1097 \newcommand*\doininsidelineRhook[1]{\gdef\doininsidelineRhook{#1}}%
1098

```

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

```

\dolineRhook 1099 \newcommand*\dolineLhook{}
\doininsidelineLhook 1100 \newcommand*\dolineRhook{}
\doininsidelineRhook 1101 \newcommand*\doininsidelineLhook{}
1102 \newcommand*\doininsidelineRhook{}
1103

```

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

```

1104 \newcommand*\do@lineR{%
1105 \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscr}%
1106 \ledRcol@true%
1107 \advance\countRline \@ne%
1108 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscr}%
1109 {\vbadness=10000%
1110 \splittopskip=\z@%
1111 \do@lineRhook%
1112 \l@demtyd@ta%
1113 \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscr}%
1114 to\baselineskip}%
1115 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startR}{}%
1116 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
1117 \getline@numR%
1118 \ifnum\@lockR>\@ne%
1119 \inserthangingsymbolRtrue%
1120 \else%
1121 \inserthangingsymbolRfalse%
1122 \fi%
1123 \setbox\l@drightbox%
1124 \hb@xt@ \Rcolwidth{%
1125 \ifl@dhidenumber%
1126 \global\l@dhidenumberfalse%
1127 \f@x@l@cksR%
1128 \else%
1129 \affixline@numR%
1130 \fi%

```



```

1131 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1132 {\add@insertsR\affixside@noteR}%
1133 {\print@lineR}%
1134 }%
1135 \add@penaltiesR%
1136 \global\advance\@donereallinesR\@ne%
1137 \global\advance\@donetotallinesR\@ne%
1138 \else%
1139 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*\Rcolwidth}%
1140 \global\advance\@donetotallinesR\@ne%
1141 \fi%
1142 \ledRcol@false%
1143 }
1144
1145

```

```

\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.

```

1146 \newcommand*{\getline@numR}{%
1147 \global\advance\absline@numR \@ne
1148 \do@actionsR
1149 \do@ballastR
1150 \ifledgroupnotesR\else
1151 \ifnumberline
1152 \ifsublines@
1153 \ifnum\sub@lockR<\tw@
1154 \global\advance\subline@numR \@ne
1155 \fi
1156 \else
1157 \ifnum\@lockR<\tw@
1158 \global\advance\line@numR \@ne
1159 \global\subline@numR \z@
1160 \fi
1161 \fi
1162 \fi
1163 \fi
1164 }
1165 \newcommand*{\getline@numL}{%
1166 \global\advance\absline@num \@ne
1167 \do@actions
1168 \do@ballast
1169 \ifledgroupnotesL\else
1170 \ifnumberline
1171 \ifsublines@
1172 \ifnum\sub@lock<\tw@

```



```

1173         \global\advance\subline@num \@ne
1174     \fi
1175     \else
1176         \ifnum\@lock<\tw@
1177             \global\advance\line@num \@ne
1178             \global\subline@num \z@
1179         \fi
1180     \fi
1181 \fi
1182 \fi
1183 }
1184
1185

```

`\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.

```

1186 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1187 \begingroup
1188     \advance\absline@numR \@ne
1189     \ifnum\next@actionlineR=\absline@numR
1190         \ifnum\next@actionR>-1001
1191             \global\advance\ballast@count by -\c@ballast
1192         \fi
1193     \fi
1194 \endgroup}

```

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

```

1195
1196 \newif\ifl@dskipversenumberR
1197 \newcommand*{\do@actions@fixedcodeR}{%
1198     \ifcase\@l@dttempcnta%
1199     \or% % 1001
1200         \global\sublines@true
1201     \or% % 1002
1202         \global\sublines@false
1203     \or% % 1003
1204         \global\@lockR=\@ne
1205     \or% % 1004%
1206         \ifnum\@lockR=\tw@
1207             \global\@lockR=\thr@@
1208         \else
1209             \global\@lockR=\z@
1210         \fi
1211     \or% % 1005
1212         \global\sub@lockR=\@ne

```



```

1213 \or% % 1006
1214 \ifnum\sub@lockR=\tw@
1215 \global\sub@lockR=\thr@@
1216 \else
1217 \global\sub@lockR=\z@
1218 \fi
1219 \or% % 1007
1220 \l@dskipnumbertrue
1221 \or% % 1008
1222 \l@dskipversenumberRtrue%
1223 \or% % 1009
1224 \l@dhidenumbertrue%
1225 \else%
1226 \led@warn@BadAction
1227 \fi%
1228 }
1229
1230
1231 \newcommand*{\do@actionsR}{%
1232 \global\let\do@actions@nextR=\relax
1233 \@l@dtmpcntb=\absline@numR
1234 \ifnum\@l@dtmpcntb<\next@actionlineR\else
1235 \ifnum\next@actionR>-1001\relax
1236 \global\page@numR=\next@actionR
1237 \ifbypage@R
1238 \global\line@numR \z@ \global\subline@numR \z@
1239 \fi
1240 \else
1241 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1242 \@l@dtmpcnta=-\next@actionR
1243 \advance\@l@dtmpcnta by -5001\relax
1244 \ifsublines@
1245 \global\subline@numR=\@l@dtmpcnta
1246 \else
1247 \global\line@numR=\@l@dtmpcnta
1248 \fi
1249 \else
1250 \@l@dtmpcnta=-\next@actionR
1251 \advance\@l@dtmpcnta by -1000\relax
1252 \do@actions@fixedcodeR
1253 \fi
1254 \fi
1255 \ifx\actionlines@listR\empty
1256 \gdef\next@actionlineR{1000000}%
1257 \else
1258 \gl@p\actionlines@listR\to\next@actionlineR
1259 \gl@p\actions@listR\to\next@actionR
1260 \global\let\do@actions@nextR=\do@actionsR
1261 \fi
1262 \fi

```



```

1263 \do@actions@nextR}
1264

```

18.4 Line number printing

\l@dcalcnun \affixline@numR is the right text version of the \affixline@num macro.

```

\ch@cksub@l@ckR 1265
\ch@ck@l@ckR 1266 \providecommand*{\l@dcalcnun}[3]{%
\fx@l@cksR 1267 \ifnum #1 > #2\relax
\affixline@numR 1268 \l@dtempcnta = #1\relax
1269 \advance\l@dtempcnta by -#2\relax
1270 \divide\l@dtempcnta by #3\relax
1271 \multiply\l@dtempcnta by #3\relax
1272 \advance\l@dtempcnta by #2\relax
1273 \else
1274 \l@dtempcnta=#2\relax
1275 \fi}
1276
1277 \newcommand*{\ch@cksub@l@ckR}{%
1278 \ifcase\sub@lockR
1279 \or
1280 \ifnum\sublock@disp=\@ne
1281 \l@dtempcntb \z@ \l@dtempcnta \@ne
1282 \fi
1283 \or
1284 \ifnum\sublock@disp=\tw@
1285 \else
1286 \l@dtempcntb \z@ \l@dtempcnta \@ne
1287 \fi
1288 \or
1289 \ifnum\sublock@disp=\z@
1290 \l@dtempcntb \z@ \l@dtempcnta \@ne
1291 \fi
1292 \fi}
1293
1294 \newcommand*{\ch@ck@l@ckR}{%
1295 \ifcase\@lockR
1296 \or
1297 \ifnum\lock@disp=\@ne
1298 \l@dtempcntb \z@ \l@dtempcnta \@ne
1299 \fi
1300 \or
1301 \ifnum\lock@disp=\tw@
1302 \else
1303 \l@dtempcntb \z@ \l@dtempcnta \@ne
1304 \fi
1305 \or
1306 \ifnum\lock@disp=\z@
1307 \l@dtempcntb \z@ \l@dtempcnta \@ne

```



```

1308   \fi
1309   \fi}
1310
1311 \newcommand*{\f@x@l@cksR}{%
1312   \ifcase\@lockR
1313   \or
1314     \global\@lockR \tw@
1315   \or \or
1316     \global\@lockR \z@
1317   \fi
1318   \ifcase\sub@lockR
1319   \or
1320     \global\sub@lockR \tw@
1321   \or \or
1322     \global\sub@lockR \z@
1323   \fi}
1324
1325
1326 \newcommand*{\affixline@numR}{%
1327   \ifledgroupnotesR\else\ifnumberline
1328   \ifl@dskipnumber
1329     \global\l@dskipnumberfalse
1330   \else
1331     \ifsublines@
1332       \@l@tempcntb=\subline@numR
1333       \l@dcalcnnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1334       \ch@cksub@lockR
1335     \else
1336       \@l@tempcntb=\line@numR
1337       \ifx\linenumberlist\empty
1338         \l@dcalcnnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1339       \else
1340         \@l@tempcnta=\line@numR
1341         \edef\rem@inder{\linenumberlist,\number\line@numR,%}
1342         \edef\sc@n@list{\def\noexpand\sc@n@list
1343           ###1,\number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1344         \sc@n@list\expandafter\sc@n@list\rem@inder|
1345         \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1346       \fi
1347       \ch@ck@l@ckR
1348     \fi
1349     \ifnum\@l@tempcnta=\@l@tempcntb
1350       \ifl@dskipversenumberR\else
1351         \if@twocolumn
1352           \if@firstcolumn
1353             \gdef\l@dld@ta{\llap{\leftlinenumR}}%
1354           \else
1355             \gdef\l@drd@ta{\rlap{\rightlinenumR}}%
1356           \fi
1357         \else

```



```

1358      \@l@dttempcntb=\line@marginR
1359      \ifnum\@l@dttempcntb>\@ne
1360        \advance\@l@dttempcntb by\page@numR
1361      \fi
1362      \ifodd\@l@dttempcntb
1363        \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1364      \else
1365        \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1366      \fi
1367    \fi
1368  \fi
1369 \fi
1370 \f@x@l@cksR
1371 \fi
1372 \fi
1373 \fi}

```

18.5 Pstart number printing in side

The printing of the pstart number is like in eledmac, 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 1374
  \leftpstartnumR 1375 \newcommand*{\affixpstart@numL}{%
\rightpstartnumR 1376 \ifsidepstartnum
  \leftpstartnumL 1377 \if@twocolumn
\rightpstartnumL 1378   \if@firstcolumn
  \ifpstartnumR 1379     \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1380     \else
1381       \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1382     \fi
1383   \else
1384     \@l@dttempcntb=\line@margin
1385     \ifnum\@l@dttempcntb>\@ne
1386       \advance\@l@dttempcntb \page@num
1387     \fi
1388     \ifodd\@l@dttempcntb
1389       \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1390     \else
1391       \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1392     \fi
1393   \fi
1394 \fi
1395 }

```



```

1396 \newcommand*{\affixpstart@numR}{%
1397 \ifsidepstartnum
1398 \if@twocolumn
1399     \if@firstcolumn
1400         \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1401     \else
1402         \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1403     \fi
1404 \else
1405     \l@dtempcntb=\line@marginR
1406     \ifnum\l@dtempcntb>\@ne
1407         \advance\l@dtempcntb \page@numR
1408     \fi
1409     \ifodd\l@dtempcntb
1410         \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1411     \else
1412         \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1413     \fi
1414 \fi
1415 \fi
1416 }
1417
1418 \newcommand*{\leftpstartnumL}{
1419 \ifpstartnum
1420 \thepstartL
1421 \kern\linenumsep\global\pstartnumfalse\fi
1422 }
1423 \newcommand*{\rightpstartnumL}{
1424 \ifpstartnum\kern\linenumsep
1425 \thepstartL
1426 \global\pstartnumfalse\fi
1427 }
1428 \newif\ifpstartnumR
1429 \pstartnumRtrue
1430 \newcommand*{\leftpstartnumR}{
1431 \ifpstartnumR
1432 \thepstartR
1433 \kern\linenumsep\global\pstartnumRfalse\fi
1434 }
1435 \newcommand*{\rightpstartnumR}{
1436 \ifpstartnumR\kern\linenumsep
1437 \thepstartR
1438 \global\pstartnumRfalse\fi
1439 }

```

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.


```

1440 \list@create{\inserts@listR}

\add@insertsR The right text version.
\add@inserts@nextR 1441 \newcommand*{\add@insertsR}{%
1442   \global\let\add@inserts@nextR=\relax
1443   \ifx\inserts@listR\empty \else
1444     \ifx\next@insertR\empty
1445       \ifx\insertlines@listR\empty
1446         \global\noteschanged@true
1447         \gdef\next@insertR{100000}%
1448       \else
1449         \gl@p\insertlines@listR\to\next@insertR
1450       \fi
1451     \fi
1452     \ifnum\next@insertR=\absline@numR
1453       \gl@p\inserts@listR\to\@insertR
1454       \@insertR
1455       \global\let\@insertR=\undefined
1456       \global\let\next@insertR=\empty
1457       \global\let\add@inserts@nextR=\add@insertsR
1458     \fi
1459   \fi
1460 \add@inserts@nextR}
1461

```

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@tempcnta` 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@tempcnta=\ballast@count
\ifnum\num@linesR>\@ne
\global\advance\par@lineR \@ne
\ifnum\par@lineR=\@ne
\advance\@l@tempcnta by \clubpenalty
\fi
\@l@tempcntb=\par@lineR \advance\@l@tempcntb \@ne
\ifnum\@l@tempcntb=\num@linesR
\advance\@l@tempcnta 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. Peter Wilson thinks 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, Peter Wilson ends up with the following.

```

1462 \newcommand*{\add@penaltiesL}{ }
1463 \newcommand*{\add@penaltiesR}{ }
1464

```

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.

```

1465 \newcommand*{\flush@notesR}{%
1466   \@xloop
1467   \ifx\inserts@listR\empty \else
1468     \glp\inserts@listR\to\@insertR
1469     \@insertR
1470     \global\let\@insertR=\undefined
1471   \repeat}
1472

```

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 21.3 p. 74 of *eledmac*’ handbook: 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:
`\printlinesR` #1 | #2 | #3 | #4 | #5 | #6 | #7
`\printlinesR` start-page | line | subline | end-page | line | subline | font

```

1473 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1474   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1475   \ifl@d@pnum #1\fullstop\fi
1476   \ifledplinenum \linenumr@p{#2}\Rlineflag\else \symplinenum\fi
1477   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1478   \ifl@d@dash \endashchar\fi
1479   \ifl@d@pnum #4\fullstop\fi
1480   \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1481   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1482 \endgroup}
1483
1484 \let\ledsavedprintlines\printlines
1485
```

19.2 Footnotes output specific to \Pages

`\print@Xnotes@forpages` The `\onlyXside` and `\onlysideX` hooks for `\Pages` allow notes to be printed
`\correct@Xfootins@box` either in left or right pages only. The implementation of such features is delegated
`\print@notesX@forpages` to `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here
`\correct@footinsX@box` 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).

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

- 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, 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`

```
1486 \newcommand\print@Xnotes@forpages[1]{%
```

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

```
1487   \ifcseempty{onlyXside@#1}{%
1488     \csuse{#1footstart}{#1}%
1489     \csuse{#1footgroup}{#1}%
1490   }%
```

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

```
1491   {%
1492     \ifboolexpr{%
1493       ((test {\ifcsstring{onlyXside@#1}{L}} and not test{\ifnumodd{\c@page}}))%
1494       or%
1495       (test {\ifcsstring{onlyXside@#1}{R}} and test{\ifnumodd{\c@page}}))%
1496     }%
```


If we are in a page where notes must be printed, print the notes, after having made the corrections which are needed for boxes.

```
1497      {%
1498          \correct@Xfootins@box{#1}%
1499          \csuse{#1footstart}{#1}%
1500          \csuse{#1footgroup}{#1}%
```

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

```
1501          \global\count\csuse{#1footins}=0%
1502          \global\skip\csuse{#1footins}=0pt%
```

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

```
1503          \csuse{Xnotefontsize@#1}%
1504          \global\advance\dimen\csuse{#1footins} by -\baselineskip%
```

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

```
1505          \global\boolfalse{keepforXside@#1}%
1506      }%
```

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.

```
1507      {%
1508          \global\booltrue{keepforXside@#1}%
```

Then restore expected rooms for notes on the next page.

```
1509          \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1510          \global\skip\csuse{#1footins}=\csuse{beforeXnotes@#1}%
```

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

```
1511          \bgroup%
1512              \csuse{Xnotefontsize@#1}%
1513              \global\advance\dimen\csuse{#1footins} by \baselineskip%
1514          \egroup%
1515 % End of \cs{print@Xnotes@forpages}.
1516     }%
1517 }%
1518 }%
```

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

```
1519 \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.

```
1520 \ifbool{keepforXside@#1}{%
```

Some setting needed to do the right splitting.

```
1521     \csuse{Xnotefontsize@#1}%
1522     \splittopskip=0pt%
```


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

```

1523     \global\setbox\csuse{#1footins}=\vbox{%
1524     \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1525     \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1526     \unvbox\csuse{#1footins}%
1527     }%

```

End of the macro.

```

1528 }{}%
1529 }%

```

And now, the same for familiar footnotes.

```

1530 \newcommand\print@notesX@forpages[1]{%
1531   \ifcempty{onlysideX@#1}{%
1532     \csuse{footstart#1}{#1}%
1533     \csuse{footgroup#1}{#1}%
1534   }%
1535   {%
1536     \ifboolexpr{%
1537       ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page}})%
1538       or%
1539       (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}}))%
1540     }%
1541     {%
1542       \correct@footinsX@box{#1}%
1543       \csuse{footstart#1}{#1}%
1544       \csuse{footgroup#1}{#1}%
1545       \global\count\csuse{footins#1}=0%
1546       \global\skip\csuse{footins#1}=0pt%
1547       \csuse{notefontsizeX@#1}%
1548       \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1549       \global\boolfalse{keepforsideX@#1}%
1550     }%
1551     {%
1552       \global\booltrue{keepforsideX@#1}%
1553       \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1554       \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1555       \bgroup%
1556         \csuse{notefontsizeX@#1}%
1557         \global\advance\dimen\csuse{footins#1} by \baselineskip%
1558       \egroup%
1559     }%
1560   }%
1561 }%
1562 \newcommand{\correct@footinsX@box}[1]{%
1563   \ifbool{keepforsideX@#1}{%
1564     \csuse{notefontsizeX@#1}%
1565     \splittopskip=0pt%
1566     \global\setbox\csuse{footins#1}=\vbox{%
1567       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%

```



```

1568      \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1569      \unvbox\csuse{footins#1}%
1570    }%
1571  }{}%
1572 }%

```

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.

```

1573 \list@create{\labelref@listR}
1574

```

`\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`.

```

1575 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1576   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1577     \led@warn@DuplicateLabel{#4}%
1578   \fi
1579   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\Rlineflag|#3|#4}%
1580   \ignorespaces}
1581 \AtBeginDocument{%
1582   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1583 }
1584

```

`\@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.

```

1585 \renewcommand*{\@lab}{%
1586   \ifledRcol
1587     \xright@appenditem{\linenumr@p{\line@numR}}{|%
1588     \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1589     \to\labelref@listR
1590   \else
1591     \xright@appenditem{\linenumr@p{\line@num}}{|%
1592     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1593     \to\labelref@list
1594   \fi}
1595

```

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* 1596 \WithSuffix\newcommand\sidenotemargin*[1]{%
1597   \l@dgetsidenote@margin{#1}
1598   \global\sidenote@marginR=\@l@dttempcntb
1599   \global\sidenote@margin=\@l@dttempcntb
1600 }
1601 \newcount\sidenote@marginR
1602 \global\sidenote@margin=\@ne
1603
```

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

```
1604 \newcommand*{\affixside@noteR}{%
1605   \def\sidenotecontent@{}%
1606   \numgdef{\itemcount@}{0}%
1607   \def\do##1{%
1608     \ifnumequal{\itemcount@}{0}%
1609     {%
1610       \appto\sidenotecontent@{##1}}% Not print not separator before the 1st note
1611       {\appto\sidenotecontent@{\sidenotesep ##1}}%
1612     }%
1613     \numgdef{\itemcount@}{\itemcount@+1}%
1614   }%
1615   \dolistloop{\l@dcnotetext}%
1616   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1617   \gdef\@templ@d{%
1618     \gdef\@templ@n{\l@dcnotetext\l@dcnotetext@l\l@dcnotetext@r}%
1619     \ifx\@templ@d\@templ@n \else%
1620       \if@twocolumn%
1621         \if@firstcolumn%
1622           \setl@dlp@rbox{##1}{\sidenotecontent@}%
1623         \else%
1624           \setl@drp@rbox{\sidenotecontent@}%
1625         \fi%
1626       \else%
1627         \@l@dttempcntb=\sidenote@marginR%
1628         \ifnum\@l@dttempcntb>\@ne%
1629           \advance\@l@dttempcntb by\page@numR%
1630         \fi%
1631         \ifodd\@l@dttempcntb%
1632           \setl@drp@rbox{\sidenotecontent@}%
1633         \gdef\sidenotecontent@{}%
1634         \numdef{\itemcount@}{0}%
1635         \dolistloop{\l@dcnotetext@l}%
1636         \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1637         \setl@dlp@rbox{\sidenotecontent@}%
1638       \else%
1639         \setl@dlp@rbox{\sidenotecontent@}%
1640         \gdef\sidenotecontent@{}%
1641         \numdef{\itemcount@}{0}%

```



```

1642      \dolistloop{\l@dcstotetext@r}%
1643      \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1644      \setl@drp@rbox{\sidenotecontent@}%
1645      \fi%
1646    \fi%
1647  \fi%
1648 }
1649

```

22 Familiar footnotes

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

```

1650 \renewcommand{\l@dbfnote}[1]{%
1651   \ifnumberedpar@
1652   \gdef\@tag{#1\relax}%
1653   \ifledRcol%
1654     \xright@appenditem{\noexpand\vl@dbfnote{\@tag}}{\@thefnmark}}%
1655     \to\inserts@listR
1656     \global\advance\insert@countR \@ne%
1657   \else%
1658     \xright@appenditem{\noexpand\vl@dbfnote{\@tag}}{\@thefnmark}}%
1659     \to\inserts@list
1660     \global\advance\insert@count \@ne%
1661   \fi
1662 \fi\ignorespaces}
1663

```

`\normalbfnoteX`

```

1664 \renewcommand{\normalbfnoteX}[2]{%
1665   \ifnumberedpar@
1666   \ifledRcol%
1667     \ifluatex
1668       \footnotelang@lua[R]%
1669     \fi
1670     \@ifundefined{xpg@main@language}%if polyglossia
1671     {}%
1672     {\footnotelang@poly[R]}%
1673     \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1674     \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}}%
1675     \to\inserts@listR
1676     \global\advance\insert@countR \@ne%
1677   \else%
1678     \ifluatex
1679       \footnotelang@lua%
1680     \fi
1681     \@ifundefined{xpg@main@language}%if polyglossia
1682     {}%

```



```

1683      {\footnotelang@poly}%
1684      \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1685      \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}}%
1686                        \to\inserts@list
1687      \global\advance\insert@count \@ne%
1688      \fi
1689      \fi\ignorespaces}
1690

```

23 Verse

Like in `eledmac`, the insertion of `hangingsymbol` is base on `\ifinserthangingsymbol`, and, for the right side, on `\ifinserthangingsymbolR`. Both commands also include the hanging space, to be sure the `\one@line` of hanging lines has the same width that the `\one@line` of normal lines and to prevent the column separator from shifting.

```

\inserthangingsymbolL
\inserthangingsymbolR 1691 \newif\ifinserthangingsymbolR
1692 \newcommand{\inserthangingsymbolL}{%
1693   \ifinserthangingsymbol%
1694     \ifinstanzaL%
1695       \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1696         \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1697       \hangingsymbol%
1698     \fi%
1699   \fi%
1700 }%
1701 \newcommand{\inserthangingsymbolR}{%
1702   \ifinserthangingsymbolR%
1703     \ifinstanzaR%
1704       \hskip \@ifundefined{sza@0@}{0}{\expandafter%
1705         \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1706       \hangingsymbol%
1707     \fi%
1708   \fi%
1709 }%

```

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.

```

1710 \chardef\next=\catcode`\&
1711 \catcode`\&=\active
1712

```

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

```

1713 \newenvironment{astanza}{%

```



```

1714 \startstanzahook
1715 \catcode`\&\active
1716 \global\stanza@count\@ne\stanza@modulo\@ne
1717 \ifnum\usernamecount{sza@00}=\z@
1718   \let\stanza@hang\relax
1719   \let\endlock\relax
1720 \else
1721   \rightskip\z@ plus 1fil\relax
1722 \fi
1723 \ifnum\usernamecount{szp@00}=\z@
1724   \let\sza@penalty\relax
1725 \fi
1726 \def&{%
1727   \endlock\mbox{}}%
1728   \sza@penalty
1729   \global\advance\stanza@count\@ne
1730   \@astanza@line}%
1731 \def\&\@stopastanza}%
1732 \pstart
1733 \@astanza@line
1734 }{}
1735

```

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

```

1736 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1737   \endlock\mbox{}}%
1738   \pend[#1]%
1739   \endstanzasextra%
1740 }%

```

`\@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.

```

1741 \newcommand*{\@astanza@line}{%
1742   \ifnum\value{stanzaindentrepetition}=0
1743     \parindent=\csname sza@\number\stanza@count
1744       @\endcsname\stanzaindentbase
1745   \else
1746     \parindent=\csname sza@\number\stanza@modulo
1747       @\endcsname\stanzaindentbase
1748     \managestanza@modulo
1749   \fi
1750   \par
1751   \stanza@hang%\mbox{}}%
1752   \ignorespaces}
1753

```

Lastly reset the modified category codes.

```

1754 \catcode`\&=\next
1755

```


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
\setnamebox the regular box macros, but including the string 'name'.
\unhnamebox 1756 \providecommand*\newnamebox}[1]{%
\unvnamebox 1757 \expandafter\newbox\csname #1\endcsname}
\namebox 1758 \providecommand*\setnamebox}[1]{%
1759 \expandafter\setbox\csname #1\endcsname}
1760 \providecommand*\unhnamebox}[1]{%
1761 \expandafter\unhbox\csname #1\endcsname}
1762 \providecommand*\unvnamebox}[1]{%
1763 \expandafter\unvbox\csname #1\endcsname}
1764 \providecommand*\namebox}[1]{%
1765 \csname #1\endcsname}
1766

\newnamecount Macros for creating and using 'named' counts.
\usenamecount 1767 \providecommand*\newnamecount}[1]{%
1768 \expandafter\newcount\csname #1\endcsname}
1769 \providecommand*\usenamecount}[1]{%
1770 \csname #1\endcsname}
1771
```

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.
1772 \newcount\l@dc@maxchunks
1773 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1774 \maxchunks{5120}
1775

\l@dnumpstartsL The numbers of left and right chunks. \l@dnumpstartsL is defined in eledmac.
\l@dnumpstartsR 1776 \newcount\l@dnumpstartsR
1777

\l@pscL A couple of scratch counts for use in left and right texts, respectively.
\l@pscR 1778 \newcount\l@dpscL
1779 \newcount\l@dpscR
1780
```


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

```

1781 \newcommand*{\l@dssetuprawboxes}{%
1782   \@l@dssetuprawboxes=\l@dc@maxchunks
1783   \loop\ifnum\@l@dssetuprawboxes>\z@
1784     \newnamebox{\l@dLcolrawbox\the\@l@dssetuprawboxes}
1785     \newnamebox{\l@dRcolrawbox\the\@l@dssetuprawboxes}
1786     \advance\@l@dssetuprawboxes \m@ne
1787   \repeat}
1788
```

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

```

1789 \newcommand*{\l@dssetupmaxlinecounts}{%
1790   \@l@dssetupmaxlinecounts=\l@dc@maxchunks
1791   \loop\ifnum\@l@dssetupmaxlinecounts>\z@
1792     \newnamecount{\l@dmaxlinesinpar\the\@l@dssetupmaxlinecounts}
1793     \advance\@l@dssetupmaxlinecounts \m@ne
1794   \repeat}
1795 \newcommand*{\l@dzeromaxlinecounts}{%
1796   \begingroup
1797   \@l@dssetupmaxlinecounts=\l@dc@maxchunks
1798   \loop\ifnum\@l@dssetupmaxlinecounts>\z@
1799     \global\usenamecount{\l@dmaxlinesinpar\the\@l@dssetupmaxlinecounts}=\z@
1800     \advance\@l@dssetupmaxlinecounts \m@ne
1801   \repeat
1802   \endgroup}
1803
```

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

```

1804 \AtBeginDocument{%
1805   \l@dssetuprawboxes
1806   \l@dssetupmaxlinecounts
1807   \l@dzeromaxlinecounts
1808   \l@dnumstartL=\z@
1809   \l@dnumstartR=\z@
1810   \l@dpscl=\z@
1811   \l@dpscr=\z@}
1812
```

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 1813 \newif\ifl@dusedbabel
\l@dusedbabeltrue
\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 \.
1814 \newcommand*{\l@dbbl@set@language}[1]{%
1815   \edef\language{#1}%
1816   \select@language{\language}%
1817   \if@filesw
1818     \protected@write\auxout{}\string\select@language{\language}%
1819     \addtocontents{toc}{\string\select@language{\language}%
1820     \addtocontents{lof}{\string\select@language{\language}%
1821     \addtocontents{lot}{\string\select@language{\language}%
1822   \fi}
1823

```

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
\l@duselanguage are the names of the languages of the left and right texts. \l@duselanguage is
\theledlanguageL similar to \selectlanguage.
\theledlanguageR 1824 \providecommand{\selectlanguage}[1]{%
1825   \newcommand*{\l@duselanguage}[1]{%
1826     \gdef\theledlanguageL{}
1827     \gdef\theledlanguageR{}
1828

```

Now do the babel fix or polyglossia, if necessary.

```

1829 \AtBeginDocument{%
1830   \ifundefined{xpg@main@language}{%
1831     \ifundefined{bbl@main@language}{%

```


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

```
1832 \l@dusedbabelfalse
1833 \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.

```
1834 \l@dusedbabeltrue
1835 \let\l@doldselectlanguage\selectlanguage
1836 \let\l@doldbbl@set@language\bbl@set@language
1837 \let\bbl@set@language\l@dbbl@set@language
1838 \renewcommand{\selectlanguage}[1]{%
1839 \l@doldselectlanguage{#1}%
1840 \ifledRcol \gdef\theledlanguageR{#1}%
1841 \else \gdef\theledlanguageL{#1}%
1842 \fi}
```

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

```
1843 \renewcommand*{\l@duselanguage}[1]{%
1844 \l@doldselectlanguage{#1}}
```

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

```
1845 \gdef\theledlanguageL{\bbl@main@language}%
1846 \gdef\theledlanguageR{\bbl@main@language}%
1847 }%
1848 }
```

If on Polyglossia

```
1849 { \let\old@otherlanguage\otherlanguage%
1850 \renewcommand{\otherlanguage}[2][]{%
1851 \selectlanguage[#1]{#2}%
1852 \ifledRcol \gdef\theledlanguageR{#2}%
1853 \else \gdef\theledlanguageL{#2}%
1854 \fi}%
1855 \let\l@duselanguage\select@language%
1856 \gdef\theledlanguageL{\xpg@main@language}%
1857 \gdef\theledlanguageR{\xpg@main@language}%

```

That's it.

```
1858 }}
```

`\if@pstarts` `\check@pstarts` returns `\@pstartstrue` if there are any unprocessed chunks.

```
\@pstartstrue 1859 \newif\if@pstarts
\@pstartsfalse 1860 \newcommand*\check@pstarts}{%
\check@pstarts 1861 \@pstartsfalse
1862 \ifnum\l@dnumpstartsL>\l@dpscl
1863 \@pstartstrue
1864 \else
1865 \ifnum\l@dnumpstartsR>\l@dpscl
1866 \@pstartstrue
```



```

1867     \fi
1868     \fi
1869 }
1870

```

`\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 sets `\araw@textfalse`.

```

\checkraw@text 1871 \newif\ifaraw@text
1872 \newcommand*{\checkraw@text}{%
1873     \araw@textfalse
1874     \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}
1875         \araw@texttrue
1876     \else
1877         \ifvbox\namebox{1@dRcolrawbox\the\1@dpscR}
1878             \araw@texttrue
1879         \fi
1880     \fi
1881 }
1882

```

`\@writelinesinparL` These write the number of text lines in a chunk to the section files, and then `\@writelinesinparR` afterwards zero the counter.

```

1883 \newcommand*{\@writelinesinparL}{%
1884     \edef\next{%
1885         \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1886     \next
1887     \global\@donereallinesL \z@}
1888 \newcommand*{\@writelinesinparR}{%
1889     \edef\next{%
1890         \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1891     \next
1892     \global\@donereallinesR \z@}
1893

```

27 Parallel columns

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

```

\@eledsectionR 1894 \newsavebox{\@eledsectionL}%
1895 \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.

```

1896 \newcommand*{\Columns}{%
1897     \if1@dpairing%
1898         \led@err@Columns@InsideEnv%

```



```

1899 \fi%
1900 \l@dprintingcolumnstrue%
1901 \eledsection@correcting@skip=-\baselineskip% Correction for sections' titles
1902 \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1903 \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
1904 \fi

```

Start a group and zero counters, etc.

```

1905 \begingroup
1906 \l@dzeropenalties
1907 \endgraf\global\l@num@lines=\prevgraf
1908 \global\l@num@linesR=\prevgraf
1909 \global\l@par@line=\z@
1910 \global\l@par@lineR=\z@
1911 \global\l@dpscL=\z@
1912 \global\l@dpscR=\z@

```

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

```

1913 \check@pstarts
1914 \loop\if@pstarts
1915 \global\l@pstartnumtrue
1916 \global\l@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.

```

1917 \global\advance\l@dpscL \@ne
1918 \global\advance\l@dpscR \@ne
1919 \restore@pstartL@pc%
1920 \restore@pstartR@pc%

```

We print the optional argument of `\pstart` or the argument of `\AtEveryPstart`.

```

1921 \Columns@print@before@pstart%

```

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

```

1922 \checkraw@text
1923 { \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.

```

1924 \l@duselanguage{\theledlanguageL}%
1925 \do@lineL
1926 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1927 {%
1928 \ifdefstring{\@eledsectmark}{L}%
1929 {\csuse{eled@sectmark@\the\l@dpscL}%
1930 }{}}%
1931 \global\csundef{eled@sectmark@\the\l@dpscL}%
1932 \savebox{\@eledsectionL}{\parbox[t]{\Lcolwidth}{\vbox{\print@eledsectionL}}}\vbox{-
1933 }%
1934 }%

```



```

1935      \l@duselanguage{\theledlanguageR}%
1936      \do@lineR
1937      \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1938      {%
1939      \ifdefstring{\@eledsectmark}{R}%
1940      {\csuse{eled@sectmark@the\l@dpscR R}%
1941      }{}}%
1942      \global\csundef{eled@sectmark@the\l@dpscR R}%
1943      \savebox{\@eledsectionR}{\parbox[t][t]{\Rcolwidth}{\vbox{}}\print@eledsectionR
1944      }{}}%
1945      \hb@xt@ \hsize{%
1946      \ifdefstring{columns@position}{L}{-}{\hfill }%
1947      \unhbox\l@dleftbox%
1948      \ifhbox\@eledsectionL%
1949      \usebox{\@eledsectionL}%
1950      \fi%
1951      \print@columnseparator%
1952      \unhbox\l@drightbox%
1953      \ifhbox\@eledsectionR%
1954      \usebox{\@eledsectionR}%
1955      \fi%
1956      \ifdefstring{columns@position}{R}{-}{\hfill}%
1957      }%
1958      \checkraw@text
1959      \checkverseL
1960      \checkverseR
1961      \checkpb@columns
1962      \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`.

```

1963      \@writelinesinparL
1964      \@writelinesinparR
1965      \check@pstarts
1966      \ifbypstart@%
1967      \write\linenum@out{\string\@set[1]}
1968      \resetprevline@
1969      \fi
1970      \ifbypstart@R
1971      \write\linenum@outR{\string\@set[1]}
1972      \resetprevline@
1973      \fi
1974      \Columns@print@after@pend%
1975      \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.

```

1976      \flush@notes
1977      \flush@notesR

```



```

1978 \endgroup
1979 \global\l@dpscL=\z@
1980 \global\l@dpscR=\z@
1981 \global\l@dnumstartsL=\z@
1982 \global\l@dnumstartsR=\z@
1983 \l@dprintingcolumnsfalse%
1984 \ignorespaces
1985 \global\instanzaLfalse
1986 \global\instanzaRfalse}
1987

```

`\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).

```

1988 \def\print@columnseparator{%
1989   \ifdim\beforecolumnseparator<0pt%
1990     \hfill%
1991   \else%
1992     \hspace{\beforecolumnseparator}%
1993   \fi%
1994   \columnseparator%
1995   \ifdim\aftercolumnseparator<0pt%
1996     \hfill%
1997   \else%
1998     \hspace{\beforecolumnseparator}%
1999   \fi%
2000 }%
2001 %\end{macrocode}
2002 % \end{macro}
2003 % \begin{macro}{\checkpb@columns}
2004 % \cs{checkpb@columns} prevent or make pagebreaking in columns, depending of the use of \cs{ledpb} or \cs{
2005 %   \begin{macrocode}
2006
2007 \newcommand{\checkpb@columns}{%
2008   \newif\if@pb
2009   \newif\if@nopb
2010   \IfStrEq{\led@pb@setting}{before}{%
2011     \numdef{\next@absline}{\the\absline@num+1}%
2012     \numdef{\next@abslineR}{\the\absline@numR+1}%
2013     \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
2014     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
2015     \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
2016     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
2017   }{}
2018   \IfStrEq{\led@pb@setting}{after}{%
2019     \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
2020     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
2021     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
2022     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%

```



```

2023   }{}
2024   \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
2025   \if@pb\pagebreak[4]\fi
2026 }

```

`\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).

```

2027 \newcommand*{\columnseparator}{%
2028   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
2029 \newdimen\columnrulewidth
2030 \columnrulewidth=\z@
2031

```

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

```

2032 \newcommand*{\columnspanposition}[1]{%
2033   \xdef\columns@position{#1}%
2034   }%
2035 \xdef\columns@position{R}%

```

`\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`.

```

2036 \newlength{\beforecolumnseparator}%
2037 \setlength{\beforecolumnseparator}{-2pt}%
2038
2039 \newlength{\aftercolumnseparator}%
2040 \setlength{\aftercolumnseparator}{-2pt}%
2041

```

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

```

setwidthliketwocolumns@R 2042 \newcommand{\setwidthliketwocolumns@L}{%
setnotepositionliketwocolumns@R 2043 % Temporary dimension, initially equal to the standard hsize, i.e. text width
2044 %   \begin{macrocode}
2045   \newdimen\temp%
2046   \temp=\hsize%

   Hsize : Left + Right width
2047   \hsize=\Lcolwidth%
2048   \advance\hsize\Rcolwidth%

```


Now, calculating the remaining space

```
2049 \advance\temp-\hsize%
```

And multiply the hsize by $2/3$ of this space

```
2050 \multiply\temp by 2%
```

```
2051 \divide\temp by 3%
```

```
2052 \advance\hsize\temp%
```

```
2053 }%
```

```
2054
```

```
2055 \newcommand{\setpositionliketwocolumns@L}{%
```

```
2056 \renewcommand{\ledrlfill}{\hfill}%
```

```
2057 }%
```

```
2058
```

```
2059 \newcommand{\setnotespositionliketwocolumns@L}{%
```

```
2060 }%
```

```
2061
```

```
2062
```

```
2063 \newcommand{\setwidthliketwocolumns@C}{%
```

```
2064 % Temporary dimension, initially equal to the standard hsize, i.e. text width
```

```
2065 \newdimen\temp%
```

```
2066 \temp=\hsize%
```

```
2067 % Hsize : Left + Right width
```

```
2068 \hsize=\Lcolwidth%
```

```
2069 \advance\hsize\Rcolwidth%
```

```
2070 % Now, calculating the remaining space
```

```
2071 \advance\temp-\hsize%
```

And multiply the hsize by $1/2$ of this space

```
2072 \divide\temp by 2%
```

```
2073 \advance\hsize\temp%
```

```
2074 }%
```

```
2075
```

```
2076 \newcommand{\setpositionliketwocolumns@C}{%
```

```
2077 \doinsidelinehook{\hfill}%
```

```
2078 \renewcommand{\ledrlfill}{\hfill}%
```

```
2079 }%
```

```
2080
```

```
2081 \newcommand{\setnotespositionliketwocolumns@C}{%
```

```
2082 \newdimen\temp%
```

```
2083 \newdimen\tempa%
```

```
2084 \temp=\hsize%
```

```
2085 \tempa=\Lcolwidth%
```

```
2086 \advance\tempa\Rcolwidth%
```

```
2087 \advance\temp-\tempa%
```

```
2088 \divide\temp by 2%
```

```
2089 \leftskip=\temp%
```

```
2090 \rightskip=-\temp%
```

```
2091 }%
```



```

2092
2093 \newcommand{\setwidthliketwocolumns@R}{%
    Temporary dimension, initially equal to the standard hsize, i.e. text width
2094   \newdimen\temp%
2095   \temp=\hsize%
    Hsize : Left + Right width
2096   \hsize=\Lcolwidth%
2097   \advance\hsize\Rcolwidth%
    Now, calculating the remaining space
2098   \advance\temp-\hsize%
    And multiply the hsize by 2/3 of this space
2099   \multiply\temp by 2%
2100   \divide\temp by 3%
2101   \advance\hsize\temp%
2102 }%
2103
2104 \newcommand{\setpositionliketwocolumns@R}{%
2105   \doinsidelinehook{\hfill}%
2106 }%
2107
2108 \newcommand{\setnotespositionliketwocolumns@R}{%
2109   \newdimen\temp%
2110   \newdimen\tempa%
2111   \temp=\hsize%
2112   \tempa=\Lcolwidth%
2113   \advance\tempa\Rcolwidth%
2114   \advance\temp-\tempa%
2115   \divide\temp by 2%
2116   \leftskip=\temp%
2117   \rightskip=-\temp%
2118 }%
2119
\Columns@print@before@pstart The \Columns@print@before@pstart and \Columns@print@after@pend print
\Columns@print@after@pend the content of the optional argument of \pstart / \pend. If this content is
not empty, it also print the separator.
2120 \newcommand{\Columns@print@before@pstart}{%
2121   \ifboolexpr{%
2122     test{\ifcsstring{before@pstartL@the\l@dpscl}{\at@every@pstart}}%
2123     and test {\ifcsstring{before@pstartR@the\l@dpscl}{\at@every@pstart}}%
2124     and test {\ifdefempty{\at@every@pstart}}}%
2125     {}%
2126     {%
2127       \hb@xt@ \hsize{%
2128         \ifdefstring{\columns@position}{L}{\hfill }%
2129         \par\parbox[t]{}[t]{\Lcolwidth}{%
2130           \csuse{before@pstartL@the\l@dpscl}%

```



```

2131         }%
2132         \print@columnseparator%
2133         \parbox[t] [] [t]{\Rcolwidth}{%
2134             \csuse{before@pstartR@the\l@dpscR}%
2135         }%
2136         \ifdefstring{\columns@position}{R}{\hfill}%
2137     }%
2138 }%
2139 \global\csundef{before@pstartL@the\l@dpscL}%
2140 \global\csundef{before@pstartR@the\l@dpscR}%
2141 }%
2142 \newcommand{\Columns@print@after@pend}{%
2143     \ifboolexpr{%
2144         test{\ifcsstring{after@pendL@the\l@dpscL}{\at@every@pend}}%
2145         and test {\ifcsstring{after@pendR@the\l@dpscR}{\at@every@pend}}%
2146         and test {\ifdefempty{\at@every@pend}}}%
2147     {%
2148     {%
2149         \hb@xt@ \hsize{%
2150             \ifdefstring{\columns@position}{L}{\hfill }%
2151             \parbox[t] [] [t]{\Lcolwidth}{%
2152                 \csuse{after@pendL@the\l@dpscL}%
2153             }%
2154             \print@columnseparator%
2155             \parbox[t] [] [t]{\Rcolwidth}{%
2156                 \csuse{after@pendR@the\l@dpscR}%
2157             }%
2158             \ifdefstring{\columns@position}{R}{\hfill}%
2159         }%
2160     }%
2161     \global\csundef{after@pendL@the\l@dpscL}%
2162     \global\csundef{after@pendR@the\l@dpscR}%
2163 }%

```

28 Parallel pages

This is considerably more complicated than parallel columns.

28.1 Specific counters

`\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` 2164 `\newcount\numpagelinesL`
2165 `\newcount\numpagelinesR`
2166 `\newcount\l@dminpagelines`
2167

28.2 Main macro

`\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.

```

2168 \newcommand*{\Pages}{%
2169   \l@dprintingpagetrue%
2170   \ifl@dpairing%
2171     \led@err@Pages@InsideEnv%
2172   \fi%
2173   \eledsection@correcting@skip=-2\baselineskip% line correcting for section titles.
2174   \parledgroup@notespacing@set@correction%
2175   \typeout{}%
2176   \typeout{***** PAGES *****}%
2177   \ifnum\l@dnumstartsL=\l@dnumstartsR\else%
2178     \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%
2179   \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.

```

2180 \setlength{\Lcolwidth}{\textwidth}%
2181 \setlength{\Rcolwidth}{\textwidth}%

```

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

```

2182 \cleartol@evenpage%
2183 \begingroup%
2184   \l@dzeropenalties%
2185   \endgraf\global\num@lines=\prevgraf%
2186   \global\num@linesR=\prevgraf%
2187   \global\par@line=\z@%
2188   \global\par@lineR=\z@%
2189   \global\l@dpscL=\z@%
2190   \global\l@dpscR=\z@%
2191   \writtenlinesLfalse%
2192   \writtenlinesRfalse%

```

Sometimes, people want to have the same page number on both left and right sides. To do this, use the `\init@sameparallepage@number` command.

```

2193   \init@sameparallepage@number

```

The footnotes are printed in a different way from expected in `eledmac`, as we may want to print the notes on one side only.

```

2194   \let\print@Xnotes\print@Xnotes@forpages%
2195   \let\print@notesX\print@notesX@forpages%

```

Check if there are chunks to be processed.

```

2196   \check@pstarts%
2197   \loop\if@pstarts%

```

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


```

2198      \global\advance\l@dpscL \@ne%
2199      \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.

```

2200      \getlinesfromparlistL%
2201      \getlinesfromparlistR%
2202      \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2203      {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
2204      \check@pstarts%
2205      \repeat%

```

Zero the counts again, ready for the next bit.

```

2206      \global\l@dpscL=\z@%
2207      \global\l@dpscR=\z@%

```

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

```

2208      \getlinesfrompagelistL%
2209      \getlinesfrompagelistR%
2210      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2211      {\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.

```

2212      \check@pstarts%
2213      \if@pstarts%

```

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

```

2214      \global\advance\l@dpscL \@ne%
2215      \global\advance\l@dpscR \@ne%
2216      \restore@pstartL@pc%
2217      \restore@pstartR@pc%

```

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

```

2218      \global\@donereallinesL=\z@%
2219      \global\@donetotallinesL=\z@%
2220      \global\@donereallinesR=\z@%
2221      \global\@donetotallinesR=\z@%

```

Start a loop over the boxes (chunks).

```

2222      \checkraw@text%

2223 %      \begingroup
2224 {      \loop\ifaraw@text%

```

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

```

2225      \checkpageL%
2226      \l@duselanguage{\theledlanguageL}%
2227 {      \loop\ifl@dsamepage%

```


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

```

2228         \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2229         \csuse{before@pstartL@the\l@dpscl}%
2230         \global\csundef{before@pstartL@the\l@dpscl}%
2231         \do@lineL%
2232         \xifinlist{the\l@dpscl}{\eled@sections@@}
2233         {\print@eledsectionL}%
2234         {}%
2235         \advance\numpagelinesL \@one%

```

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

```

2236         \ifshiftedpstarts%
2237             \ifdim\ht\l@dleftbox>Opt\hb@xt@%
2238                 \hsize{\ledstrutL\unhbox\l@dleftbox}%
2239             \else%
2240                 \dimen0=\pagetotal%
2241                 \advance\dimen0 by \baselineskip%
2242                 \global\pagetotal=\dimen0%
2243             \fi%
2244         \else%
2245             \parledgroup@correction@notespacing{L}
2246             \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2247         \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. Reset the verse skipnumber boolean

```

2248         \get@nextboxL%
2249         \global\l@dskipversenumberfalse%
2250         \ifprint@last@after@pendL%
2251             \csuse{after@pendL@the\l@dpscl}%
2252             \global\csundef{after@pendL@the\l@dpscl}%
2253         \fi%
2254         \checkpageL%
2255         \checkverseL%
2256         \checkpbl%
2257         \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.

```

2258      \ifl@dpagfull%
2259      \@writelinesonpageL{\the\numpagelinesL}%
2260      \else%
2261      \@writelinesonpageL{1000}%
2262      \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.

```

2263      \numpagelinesL \z@%
2264      \parledgroup@correction@notespacing@init%
2265      \clearl@dleftpage }%

```

Now do the same for the right text.

```

2266      \checkpageR%
2267      \l@duselanguage{\theledlanguageR}%
2268 {
2269      \loop\ifl@dsamepage%
2270      \initnumbering@sectcountR%
2271      \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2272      \csuse{before@pstartR@the\l@dpscR}%
2273      \global\csundef{before@pstartR@the\l@dpscR}%
2274      \do@lineR%
2275      \xifinlist{\the\l@dpscR}{\eled@sectionsR@}%
2276      {\print@eledsectionR}%
2277      {}%
2278      \advance\numpagelinesR \@ne%
2279      \ifshiftedpstarts%
2280      \ifdim\ht\l@drightbox>0pt\hb@xt@%
2281      \hsize{\ledstrutR\unhbox\l@drightbox}%
2282      \else%
2283      \dimen0=\pagetotal%
2284      \advance\dimen0 by \baselineskip%
2285      \global\pagetotal=\dimen0%
2286      \fi%
2287      \else%
2288      \parledgroup@correction@notespacing{R}%
2289      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2290      \fi%
2291      \get@nextboxR%
2292      \global\l@dskipversenumberRfalse%
2293      \ifprint@last@after@pendR%
2294      \csuse{after@pendR@the\l@dpscR}%
2295      \global\csundef{after@pendR@the\l@dpscR}%
2296      \fi%
2297      \checkpageR%
2298      \checkverseR%
2299      \checkpbR%
2300      \repeat%

```



```

2300      \ifl@dpagfull%
2301          \@writelinesonpageR{\the\numpagelinesR}%
2302      \else%
2303          \@writelinesonpageR{1000}%
2304      \fi%
2305      \numpagelinesR=\z@%
2306      \parledgroup@correction@notespacing@init%

```

The page is full, so move onto the next (left, odd) page and repeat left text processing.

```

2307      \clearl@drighthpage}%
      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.
2308      \checkraw@text%
2309      \ifaraw@text%
2310          \getlinesfrompagelistL%
2311          \getlinesfrompagelistR%
2312          \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2313                          {\l@dminpagelines}%
2314      \fi%
2315      \repeat}%

```

We have now output the text from all the chunks.

```

2316      \fi%
      Make sure that there are no inserts hanging around.
2317      \flush@notes%
2318      \flush@notesR%
2319      \endgroup%

```

Zero counts ready for the next set of left/right text chunks. The boolean tests for stanza are switched to false.

```

2320      \global\l@dpscL=\z@%
2321      \global\l@dpscR=\z@%
2322      \global\l@dnumstartsL=\z@%
2323      \global\l@dnumstartsR=\z@%
2324      \global\instanzaLfalse%
2325      \global\instanzaRfalse%
2326      \l@dprintingpagesfalse%
2327      \finish@sameparallepage@number%In order to have continuous page number
2328      \finish@Pages@notes%Needed to prevent final notes overlap line number
2329      \ignorespaces}
2330
2331

```

28.3 Ensure all notes be printed at the end of parallel pages

`\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 left for the next pages.

```

2332 \newcommand{\finish@Pages@notes}{%

```



```
2333 \def\do##1{%
```

First, declare footnote box if there was no previous declared. E.g. if familiar or critical notes were disabled by `eledmac` options.

```
2334 \ifnocritical{%
2335   \global\newnamebox{##1footins}
2336 \fi
2337 \ifnofamiliar{%
2338   \global\newnamebox{footins##1}
2339 \fi
```

And now, add a `\newpage` if there is no more footnote to print.

```
2340 \ifvoid\csuse{##1footins}%
2341   \ifvoid\csuse{footins##1}\else%
2342     \newpage\null%
2343     \listbreak%
2344   \fi%
2345 \else%
2346   \newpage\null%
2347   \listbreak%
2348 \fi%
2349 }%
2350 \dolistloop{\@series}%
2351 }%
```

28.4 Struts

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

```
\ledstrutR 2352 \newcommand*{\ledstrutL}{\strut}
2353 \newcommand*{\ledstrutR}{\strut}
2354
```

28.5 Page clearing

`\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.

```
2355 \providecommand{\cleartoevenpage}[1][\@empty]{%
2356   \clearpage
2357   \ifodd\c@page\hbox{##1}\clearpage\fi}
2358 \newcommand*{\cleartol@devenpage}{%
2359   \ifdim\pagetotal<\topskip% on an empty page
2360   \else
2361     \clearpage
2362   \fi
2363   \ifodd\c@page\hbox{##1}\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`

```

2364 \newcommand*{\clearl@dleftpage}{%
2365   \ifdim\pagetotal=0pt\hbox{}\fi%
2366   \newpage%
2367   \ifodd\c@page\else
2368     \led@err@LeftOnRightPage
2369     \hbox{}%
2370     \cleardoublepage
2371   \fi}
2372
2373 \newcommand*{\clearl@drightpage}{%
2374   \ifdim\pagetotal=0pt\hbox{}\fi%
2375   \newpage%
2376   \stepcounter{sameparallepage@number}%
2377   \ifodd\c@page
2378     \led@err@RightOnLeftPage
2379     \hbox{}%
2380     \cleartoevenpage
2381   \fi}
2382

```

28.6 Lines managing

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

```

\cs@linesinparR 2383 \newcommand*{\getlinesfromparlistL}{%
2384   \ifx\linesinpar@listL\empty
2385     \gdef\cs@linesinparL{0}%
2386   \else
2387     \gl@p\linesinpar@listL\to\cs@linesinparL
2388   \fi}
2389 \newcommand*{\getlinesfromparlistR}{%
2390   \ifx\linesinpar@listR\empty
2391     \gdef\cs@linesinparR{0}%
2392   \else
2393     \gl@p\linesinpar@listR\to\cs@linesinparR
2394   \fi}
2395

```

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

```

\cs@linesonpageR 2396 \newcommand*{\getlinesfrompagelistL}{%
2397   \ifx\linesonpage@listL\empty

```



```

2398 \gdef\@cs@linesonpageL{1000}%
2399 \else
2400 \gl@p\linesonpage@listL\to\@cs@linesonpageL
2401 \fi}
2402 \newcommand*\getlinesfrompagelistR{%
2403 \ifx\linesonpage@listR\empty
2404 \gdef\@cs@linesonpageR{1000}%
2405 \else
2406 \gl@p\linesonpage@listR\to\@cs@linesonpageR
2407 \fi}
2408

```

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

```

2409 \newcommand*\@writelinesonpageL[1]{%
2410 \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2411 \next}
2412 \newcommand*\@writelinesonpageR[1]{%
2413 \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2414 \next}
2415

```

\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>.

```

2416 \newcommand*\l@dcalc@maxoftwo[3]{%
2417 \ifnum #2>#1\relax
2418 #3=#2\relax
2419 \else
2420 #3=#1\relax
2421 \fi}
2422 \newcommand*\l@dcalc@minoftwo[3]{%
2423 \ifnum #2<#1\relax
2424 #3=#2\relax
2425 \else
2426 #3=#1\relax
2427 \fi}
2428

```

28.7 Page break managing

\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
 \checkpageR


```

2429 \newif\ifl@dsamepage
2430 \l@dsamepagetrue
2431 \newif\ifl@dpagefull
2432
2433 \newcommand*{\checkpageL}{%
2434   \l@dpagefulltrue
2435   \l@dsamepagetrue
2436   \check@goal
2437   \ifdim\pagetotal<\ledthegoal
2438     \ifnum\numpagelinesL<\l@dminpagelines
2439       \else
2440         \l@dsamepagefalse
2441         \l@dpagefullfalse
2442       \fi
2443     \else
2444       \l@dsamepagefalse
2445       \l@dpagefulltrue
2446     \fi%
2447   \ifprint@last@after@pendL%
2448     \l@dpagefullfalse%
2449     \l@dsamepagefalse%
2450     \print@last@after@pendLfalse%
2451   \fi%
2452 }%
2453
2454 \newcommand*{\checkpageR}{%
2455   \l@dpagefulltrue
2456   \l@dsamepagetrue
2457   \check@goal
2458   \ifdim\pagetotal<\ledthegoal
2459     \ifnum\numpagelinesR<\l@dminpagelines
2460       \else
2461         \l@dsamepagefalse
2462         \l@dpagefullfalse
2463       \fi
2464     \else
2465       \l@dsamepagefalse
2466       \l@dpagefulltrue
2467     \fi%
2468   \ifprint@last@after@pendR%
2469     \l@dpagefullfalse%
2470     \l@dsamepagefalse%
2471     \print@last@after@pendRfalse%
2472   \fi%
2473 }%
2474

```

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


```

\lednopb.
2475 \newcommand{\checkpbL}{
2476   \IfStrEq{\led@pb@setting}{after}{
2477     \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{}
2478     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{}
2479   }{}
2480   \IfStrEq{\led@pb@setting}{before}{
2481     \numdef{\next@absline}{\the\absline@num+1}
2482     \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{}
2483     \xifinlistcs{\next@absline}{l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{}
2484   }{}
2485 }
2486
2487 \newcommand{\checkpbR}{
2488   \IfStrEq{\led@pb@setting}{after}{
2489     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@dpagfulltrue\l@dsamepagefalse}{}
2490     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@dpagfullfalse\l@dsamepagetrue}{}
2491   }{}
2492   \IfStrEq{\led@pb@setting}{before}{
2493     \numdef{\next@abslineR}{\the\absline@numR+1}
2494     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@dpagfulltrue\l@dsamepagefalse}{}
2495     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@dpagfullfalse\l@dsamepagetrue}{}
2496   }{}
2497 }

```

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

```

2498 \newcommand{\checkverseL}{
2499   \ifinstanzaL
2500     \iflednopbinverse
2501       \ifinserthangingsymbol
2502         \numgdef{\prev@abslineverse}{\the\absline@num-1}
2503         \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{}
2504         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\prev@abslineverse}\fi}{}
2505       \fi
2506     \fi
2507   \fi
2508 }
2509 \newcommand{\checkverseR}{
2510   \ifinstanzaR
2511     \iflednopbinverse
2512       \ifinserthangingsymbolR
2513         \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2514         \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{}
2515         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\prev@abslineverse}\fi}{}
2516       \fi
2517     \fi
2518   \fi
2519 }

```


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

```
2520 \newdimen\ledthegoal
2521 \ifshiftedpstarts
2522     \newcommand*{\goalfraction}{0.95}
2523 \else
2524     \newcommand*{\goalfraction}{0.9}
2525 \fi
2526
2527 \newcommand*{\check@goal}{%
2528     \ledthegoal=\goalfraction\pagegoal}
2529
```

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

```
\ifwrittenlinesL 2530 \newif\ifwrittenlinesL
2531 \newif\ifwrittenlinesR
2532
```

28.8 Getting boxes content

`\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.

```
2533 \newcommand*{\get@nextboxL}{%
2534     \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}% box is not empty
```

The current box is not empty; do nothing.

```
2535 \else%                                box is empty
```

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

```
2536     \ifnum\usenamecount{1@dmaxlinesinpar\the\1@dpscL}>\@donetotallinesL
2537         \parledgroup@notes@endL
2538     \else
```

Sufficient lines have been output.

```
2539         \ifnum\usenamecount{1@dmaxlinesinpar\the\1@dpscL}=\@donetotallinesL
2540             \parledgroup@notes@endL
2541         \fi
2542     \ifwrittenlinesL\else
```

Write out the number of lines done, and set the boolean so this is only done once.

```
2543         \@writelinesinparL
2544         \writtenlinesLtrue
2545     \fi
2546     \ifnum\1@dnumpstartsL>\1@dpscL
```

There are still unprocessed boxes. Recalculate the maximum number of lines needed, and move onto the next box (by incrementing `\1@dpscL`). If needed, restart the line numbering.

```
2547         \writtenlinesLfalse
```



```

2548         \ifbypstart@
2549             \global\line@num=0%
2550             \resetprevline%
2551         \fi
2552 % Add the content of the optional argument of the previous \cs{pend}.
2553 %     \begin{macrocode}
2554         \csuse{after@pendL@the\l@dpscl}%
2555         \global\csundef{after@pendL@the\l@dpscl}%
2556
2557     Check the number of lines
2558
2559     \l@dcalcm@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2560     {\the\@donetotallinesL}%
2561     {\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2562     \global\@donetotallinesL \z@
2563
2564     Go to the next pstart
2565
2566     \global\advance\l@dpscl \@ne
2567     \global\pstartnumtrue%
2568     \restore@pstartL@pc%
2569
2570     Add notes of parallel ledgroup.
2571
2572     \parledgroup@notes@endL
2573     \parledgroup@correction@notespacing@final{L}
2574     \else
2575     \fi
2576     \fi
2577     \fi}
2578
2579 \newcommand*{\get@nextboxR}{%
2580 \ifvbox\namebox{l@dRcolrawbox\the\l@dpscl}% box is not empty
2581 \else% box is empty
2582 \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesR
2583 \parledgroup@notes@endR
2584 \else
2585 \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscl}=\@donetotallinesR
2586 \parledgroup@notes@endR
2587 \fi
2588 \ifwrittenlinesR\else
2589 \@writelinesinparR
2590 \writtenlinesRtrue
2591 \fi
2592 \ifnum\l@dnumpstartsR>\l@dpscl
2593 \writtenlinesRfalse
2594 \ifbypstart@R
2595 \global\line@numR=0%
2596 \resetprevline%
2597 \fi
2598 \csuse{after@pendR@the\l@dpscl}%
2599 \global\csundef{after@pendR@the\l@dpscl}%
2600 \l@dcalcm@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2601 {\the\@donetotallinesR}%

```



```

2592                                {\usernamecount{l@dmxlinesinpar\the\l@dpscR}}}%
2593                                \global\@donetotallinesR \z@
2594                                \global\advance\l@dpscR \@ne
2595                                \global\pstartnumRtrue%
2596                                \restore@pstartR@pc%
2597                                \parledgroup@notes@endR
2598                                \parledgroup@correction@notes@spacing@final{R}
2599                                \else
2600                                    \print@last@after@pendRtrue%
2601                                \fi
2602                                \fi
2603                                \fi}
2604

```

28.9 Same page number in both side

The `sameparallelpagenumber` allow to have the same page number for the left and the right side. We can not do it by changing the value of the `page` counter, since its value is used to determine whether a page is left or right. Consequently, we have to do it by patching `\thepage` inside a `\Pages` macro.

`\init@sameparallelpagenumber` This macro is called at the beginning of `\Pages`. It patches the `\thepage` macro in order to and to use the value of `sameparallelpagenumber` `LATEX`counter instead of those of `page` `LATEX`counter. As we are inside a group, the patch is local, and, consequently, the page printed after the `\Pages` will use the normal page number scheme.

The value of `sameparallelpagenumber` is increase by 1 when we change from right page to left page.

```

2605 \newcounter{sameparallelpagenumber}
2606 \newcommand{\init@sameparallelpagenumber}{%
2607     \setcounter{sameparallelpagenumber}{\c@page}%
2608     \ifsameparallelpagenumber%
2609         \patchcmd{\thepage}{page}{sameparallelpagenumber}{}{}%
2610     \fi%
2611 }%

```

`\finish@sameparallelpagenumber` This macro is called at the end of `\Pages`. If the `sameparallelpagenumber` is enabled, it set the page number to the last value of `sameparallelpagenumber` counter, in order to have a continuity of page numbering between pages printed with `\Pages` and normal pages.

```

2612 \newcommand{\finish@sameparallelpagenumber}{%
2613     \ifsameparallelpagenumber%
2614         \setcounter{page}{\c@sameparallelpagenumber}%
2615     \fi%
2616 }%
2617 % \end{macrocode}
2618 % \end{macro}
2619 % \section{Sections' titles' commands}

```



```

2620 % As switching from left to right pages does not clear the page since v1.13.0,
2621 % but only creates new pages, no \verb+\vbox{ }+ is inserted, and consequently parallel chapters are mis-al
2622 %
2623 % So we patch the \cs{chapter} command in order to prevent this problem.
2624 % \begin{macro}{\chapter}
2625 %   \begin{macrocode}
2626 \pretocmd{\chapter}{%
2627   \ifl@ndprintingpages%
2628     \vbox{ }%
2629   \fi%
2630 }%
2631 {}%
2632 {}%

```

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

```

2633 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2634 \eledsectnotoc{R}

```

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

```

2635 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2636 \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`.

```

2637 \newskip\eledsection@correcting@skip

```

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

```

2638 \newwrite\eled@sectioningR@out

```

29 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).

```

2639 \def\l@prev@pbR{}
2640 \def\l@prev@nopbR{}

```

`\ledpbR` The `\ledpbR` macro writes the call to `\led@pbR` in line-list file. The `\ledpbnR` macro writes the call to `\led@pbnR` in line-list file. The `\lednopbR` macro writes the call to `\led@nopbR` in line-list file.

the call to `\led@nopbR` in line-list file. The `\lednopbnumR` macro writes the call to `\led@nopbnumR` in line-list file.

```
2641 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2642 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2643 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2644 \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` `\led@pbnumR` add the argument in the `\prev@pbR` list. The `\led@nopbR` add
`\led@nopbR` the absolute line number in the `\prev@nopbR` list. The `\led@nopbnumR` add the
`\led@nopbnumR` argument in the `\prev@nopbR` list.

```
2645 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2646 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}
2647 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2648 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}
```

30 Parallel ledgroup

`\parledgroup@` The marks `\parledgroup` contains information about the beginnings and endings
`\parledgroupseries@` of notes in a parallel ledgroup. `\parledgroupseries` contains the footnote series.
`\parledgroupstype@` `\parledgroupseries` contains the type of the footnote: critical (Xfootnote) or
familiar (footnoteX).

```
2649 \newmarks\parledgroup@
2650 \newmarks\parledgroup@series
2651 \newmarks\parledgroup@type
```

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

```
2652 \newcommand{\parledgroup@notes@startL}{%
2653   \ifnum\usenamecount{\l@maxlinesinpar\the\l@dpscL}>0%
2654   \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmar
2655   \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmar
2656   \fi%
2657   \global\ledgroupnotesL@true%
2658   \insert@noterule@ledgroup{L}%
2659 }
2660 \newcommand{\parledgroup@notes@startR}{%
2661   \ifnum\usenamecount{\l@maxlinesinpar\the\l@dpscR}>0%
2662   \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmar
2663   \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmar
2664   \fi%
2665   \global\ledgroupnotesR@true%
2666   \insert@noterule@ledgroup{R}%
2667 }
```

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


```

2668 \newcommand{\parledgroup@notes@endL}{\%
2669   \global\ledgroupnotesL=false%
2670 }
2671 \newcommand{\parledgroup@notes@endR}{\%
2672   \global\ledgroupnotesR=false%
2673 }

```

`\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`

```

2674 \newcommand{\insert@noterule@ledgroup}[1]{
2675   \IfStrEq{\splitbotmarks\parledgroup@}{begin}{\%
2676     \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2677       \csuse{ifledgroupnotes#1@}
2678       \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2679       \csuse{\splitbotmarks\parledgroup@series footnoterule}
2680     \fi
2681   }
2682   {}
2683   \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2684     \csuse{ifledgroupnotes#1@}
2685     \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2686     \csuse{footnoterule\splitbotmarks\parledgroup@series}
2687     \fi
2688   }{}
2689 }
2690 {}
2691 }

```

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

```

2692 \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 begining of `\Pages`.

```

2693 \dimdef{\parledgroup@notespacing@correction}{0pt}
2694 \newcommand{\parledgroup@notespacing@set@correction}{\%
2695   {\notefontsetup\parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}\%
2696   \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing}\%
2697 }

```

`\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 begining of each pages AND at the end of each ledgroup.

```

2698 \newcommand{\parledgroup@correction@notespacing@init}{
2699   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2700   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2701 }

```


2702 `\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`.

```

2703 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2704   \ifparledgroup
2705     \vspace{\parledgroup@notespacing@correction@accumulated}
2706     \parledgroup@correction@notespacing@init%
2707     \ifstrequal{#1}{L}{
2708       \numdef{\@checking}{\the\l@dpscL-1}
2709     }{
2710       \numdef{\@checking}{\the\l@dpscR-1}
2711     }
2712     \dimdef{\@beforenotes@current@diff}{\csuse{@parledgroup@beforenotes@\@checking L}-\csus
2713     \ifstrequal{#1}{L}%
2714     {% Left
2715       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\@beforenotes@current@diff}}
2716     }%
2717     {% Right
2718       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\@beforenotes@current@diff}}
2719     }%
2720   \fi
2721 }
```

`\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.

```

2722 {}
2723 \newcommand{\parledgroup@correction@notespacing}[1]{%
2724   \csuse{ifledgroupnotes#1@}%
2725   \vspace{-\parledgroup@notespacing@correction}%
2726   \dimdef{\parledgroup@notespacing@correction@accumulated}{\parledgroup@notespacing@correction@accumulated}
2727   \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo}
2728   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip}{\advance\parledgroup@notespacing@correction@modulo\baselineskip}
2729   \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo}
2730   }% mean greater than equal
```



```

2731     \fi%
2732 }

```

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

```

2733 \dimdef\parledgroup@beforenotesL{Opt}
2734 \dimdef\parledgroup@beforenotesR{Opt}

```

`\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.

```

2735 \newcommand{\parledgroup@beforenotes@save}[1]{
2736   \ifparledgroup
2737     \csdimgdef{@parledgroup@beforenotes@\the\csuse{1@dnumpstarts#1}#1}{\csuse{parledgroup@beforenotes#
2738     \csdimgdef{parledgroup@beforenotes#1}{Opt}
2739   \fi
2740 }

```

31 The End

</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 `\setstanzaindents` 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>\&</code>	1710, 1711, 1715, 1731, 1754
<code>\@RTLtrue</code>	1078
<code>\@adv</code>	<u>379</u> , 648, 649
<code>\@afterindentfalse</code>	796
<code>\@arabic</code>	235, 236, 850, 852

\@astanza@line	1730, 1733, <u>1741</u>
\@auxout	1818
\@beforenotes@current@diff	2712, 2715, 2718
\@chapter	797
\@checking	2708, 2710, 2712
\@cs@linesinparL	2202, <u>2383</u>
\@cs@linesinparR	2202, <u>2383</u>
\@cs@linesonpageL	2210, 2312, <u>2396</u>
\@cs@linesonpageR	2210, 2312, <u>2396</u>
\@currentlabel	897, 946
\@donereallinesL	<u>1008</u> , 1043, 1885, 1887, 2218
\@donereallinesR	<u>1008</u> , 1136, 1890, 1892, 2220
\@donetotallinesL	<u>1008</u> , 1044, 1047, 2219, 2536, 2539, 2557, 2559
\@donetotallinesR	<u>1008</u> , 1137, 1140, 2221, 2572, 2575, 2591, 2593
\@edtext@level	552, 577–579, 584–588, 590, 592
\@eled@sectioningfalse	1082
\@eled@sectioningtrue	1072
\@eledsectionL	<u>1894</u> , 1932, 1948, 1949
\@eledsectionR	<u>1894</u> , 1943, 1953, 1954
\@eledsectmark	1069, 1928, 1939, 2635
\@eledsectnotoc	1068, 2228, 2270, 2633
\@gobble	560
\@gobblethree	566
\@insertR	1453–1455, 1468–1470
\@l@dttempcnta	452, 454, 456, 457, 461, 463, 465, 466, 1198, 1242, 1243, 1245, 1247, 1250, 1251, 1268–1272, 1274, 1281, 1286, 1290, 1298, 1303, 1307, 1340, 1343, 1345, 1349
\@l@dttempcntb	191, 193, 195, 1233, 1234, 1281, 1286, 1290, 1298, 1303, 1307, 1332, 1336, 1349, 1358–1360, 1362, 1384–1386, 1388, 1405–1407, 1409, 1598, 1599, 1627–1629, 1631, 1782–1786, 1790–1793, 1797–1800
\@lab	564, <u>1585</u>
\@lemma	565
\@lock	1026, 1176
\@lockR	59, 323, 325, 327, 340, 486, 502, 503, 505, 506, 534, 535, 537, 1118, 1157, 1204, 1206, 1207, 1209, 1295, 1312, 1314, 1316
\@lopL	<u>605</u> , 2410
\@lopR	560, <u>605</u> , 2413
\@nl	<u>301</u> , 631, 633
\@nl@reg	350
\@nl@regR	<u>301</u>
\@nobreakfalse	858, 908, 1071
\@nobreaktrue	856, 860, 906, 910, 1071
\@nopbtrue	2015, 2016, 2021, 2022
\@oldnobreak	856, 858, 906, 908, 964, 985
\@pbtrue	2013, 2014, 2019, 2020
\@pend	<u>596</u> , 1885
\@pendR	<u>596</u> , 1890
\@pstartsfalse	<u>1859</u>
\@pstartstrue	<u>1859</u>
\@ref	<u>549</u>

\@ref@reg	594
\@schapter	797
\@series	742, 2350
\@set	<u>411</u> , 655, 656, 1967, 1971
\@startstanza	812, 813, 838, 839
\@stopastanza	1731, <u>1736</u>
\@sw	566
\@tag	1652, 1654, 1658
\@temp	759, 760, 765, 766
\@templ@d	1617, 1619
\@templ@n	1618, 1619
\@tmp	587, 589, 590, 714, 716, 717, 720–722
\@tmpa	715, 716, 721, 724
\@tmpp	588, 589
\@writelinesinparL	<u>1883</u> , 1963, 2543
\@writelinesinparR	<u>1883</u> , 1964, 2579
\@writelinesonpageL	2259, 2261, <u>2409</u>
\@writelinesonpageR	2301, 2303, <u>2409</u>
\@xloop	1466

A

\absline@num	375, 445, 459, 478, 1166, 2011, 2019, 2021, 2477, 2478, 2481, 2502
\absline@numR	54, <u>252</u> , 303, 306, 309, 442, 450, 471, 490, 524, 555, 1147, 1188, 1189, 1233, 1452, 2012, 2020, 2022, 2489, 2490, 2493, 2513, 2645, 2647
\actionlines@list	293, 296, 445, 459, 478
\actionlines@listR	<u>256</u> , 271, 285, 288, 442, 450, 471, 490, 524, 1255, 1258
\actions@list	297, 446, 466, 480, 482
\actions@listR	<u>256</u> , 272, 289, 443, 457, 473, 475, 492, 501, 526, 533, 1259
\add@inserts	1040, 1054
\add@inserts@nextR	<u>1441</u>
\add@insertsR	1132, <u>1441</u>
\add@penaltiesL	1042, <u>1462</u>
\add@penaltiesR	1135, <u>1462</u>
\addtocontents	1819–1821
\addtocounter	967, 988, 1067
\advanceline	<u>647</u>
\affixline@num	1037
\affixline@numR	1129, <u>1265</u>
\affixpstart@numL	1052, <u>1374</u>
\affixpstart@numR	<u>1374</u>
\affixside@note	1040, 1054
\affixside@noteR	1132, <u>1604</u>
\aftercolumnseparator	4, 1995, <u>2036</u>
\appto	1610, 1611
\araw@textfalse	<u>1871</u>
\araw@texttrue	<u>1871</u>
astanza (environment)	11, <u>1713</u>
\at@begin@pairs	773, 778, 779
\at@every@pend	971, 992, 2144–2146
\at@every@pstart	900, 949, 2122–2124

\at@every@pstart@call	902, 951, 996, 997
\AtBeginDocument	1581, 1804, 1829
\AtBeginPairs	4, <u>778</u>
\AtEveryPstartCall	<u>996</u>

B

\ballast@count	1186, 1191
\bbl@main@language	1845, 1846
\bbl@set@language	1836, 1837
\beforecolumnseparator	4, 1989, 1992, 1998, <u>2036</u>
\beginnumbering	9, 826, 868
\beginnumberingR	<u>45</u> , 129, 826, 918
\bfseries	850, 852
\bodydir	864, 914, 1077
\boolfalse	579, 1505, 1549
\booltrue	1508, 1552
\bypage@Rfalse	<u>149</u> , 165, 173
\bypage@Rtrue	<u>149</u> , 157
\bypstart@Rfalse	<u>149</u> , 158, 174
\bypstart@Rtrue	<u>149</u> , 166

C

\c@ballast	1191
\c@chapter	110
\c@chapterR	110
\c@firstlinenumR	<u>200</u> , 1338
\c@firstsublinenumR	<u>204</u> , 1333
\c@linenumincrementR	<u>200</u> , 1338
\c@page	631, 633, 1493, 1495, 1537, 1539, 2357, 2363, 2367, 2377, 2607
\c@pstartL	748, 760, 850
\c@pstartR	753, 766, 852
\c@sameparallelpage@number	2614
\c@section	111
\c@sectionR	111
\c@sublinenumincrementR	<u>204</u> , 1333
\c@subsection	112
\c@subsectionR	112
\c@subsubsection	113
\c@subsubsectionR	113
\ch@ck@l@ckR	<u>1265</u>
\ch@cksub@l@ckR	<u>1265</u>
\ch@cksub@lockR	1334
\chapter	782, 783, 792, 2624, 2626
\chapterinpages	<u>769</u> , 783, 794
\chardef	1710
\check@goal	2436, 2457, <u>2520</u>
\check@pstarts	<u>1859</u> , 1913, 1965, 2196, 2204, 2212
\checkpageL	2225, 2254, <u>2429</u>
\checkpageR	2266, 2296, <u>2429</u>
\checkpb@columns	1961, 2003, 2007

<code>\checkpbL</code>	2256, 2475
<code>\checkpbR</code>	2298, 2475
<code>\checkraw@text</code>	1871 , 1922, 1958, 2222, 2308
<code>\checkverseL</code>	1959, 2255, 2498
<code>\checkverseR</code>	1960, 2297, 2498
<code>\cleardoublepage</code>	2370
<code>\clearl@leftpage</code>	2265, 2364
<code>\clearl@rightpage</code>	2307, 2364
<code>\cleartoevenpage</code>	2355 , 2380
<code>\cleartol@evenpage</code>	2182, 2355
<code>\closeout</code>	101, 618, 626
<code>\columnrulewidth</code>	4, 2027
<code>\Columns</code>	3, 33, 35, 39, 1896
<code>\columns@position</code>	1946, 1956, 2032 , 2128, 2136, 2150, 2158
<code>\Columns@print@after@pend</code>	1974, 2120
<code>\Columns@print@before@pstart</code>	1921, 2120
<code>\columnseparator</code>	4, 1994, 2027
<code>\columnspan</code>	4, 2032
<code>\correct@footinsX@box</code>	1486
<code>\correct@Xfootins@box</code>	1486
<code>\count</code>	1501, 1509, 1545, 1553
<code>\countLline</code>	1003 , 1015
<code>\countRline</code>	1003 , 1107
<code>\create@this@edtext@level</code>	585, 586
<code>\critext</code>	676
<code>\cs</code>	581, 582, 1515, 2004, 2552, 2623
<code>\csdimgdef</code>	2737, 2738
<code>\csgdef</code>	699, 704, 900, 901, 949, 950, 971, 972, 992, 993
<code>\cslet</code>	590, 717, 722, 749, 754
<code>\csundef</code>	1083, 1931, 1942, 2139, 2140, 2161, 2162, 2230, 2252, 2272, 2294, 2555, 2589
<code>\csuse</code>	712, 715, 725, 726, 1080, 1488, 1489, 1499–1504, 1509, 1510, 1512, 1513, 1521, 1523, 1524, 1526, 1532, 1533, 1543–1548, 1553, 1554, 1556, 1557, 1564, 1566, 1567, 1569, 1673, 1684, 1929, 1940, 2130, 2134, 2152, 2156, 2229, 2251, 2271, 2293, 2340, 2341, 2554, 2588, 2654, 2655, 2662, 2663, 2677–2679, 2684–2686, 2712, 2724, 2737

D

<code>\DeclareOption</code>	9–12, 14
<code>\def@tempb</code>	171
<code>\dimdef</code>	2693, 2699, 2700, 2712, 2726, 2727, 2729, 2733, 2734
<code>\dimen</code>	634, 635, 639–641, 645, 1504, 1513, 1548, 1557, 2240–2242, 2282–2284
<code>\dimexpr</code>	1524, 1525, 1567, 1568
<code>\dingdef</code>	2695, 2696
<code>\divide</code>	1270, 2051, 2072, 2088, 2100, 2115
<code>\do@actions</code>	1167
<code>\do@actions@fixedcodeR</code>	1195
<code>\do@actions@nextR</code>	1195
<code>\do@actionsR</code>	1148, 1195
<code>\do@ballast</code>	1168
<code>\do@ballastR</code>	1149, 1186

<code>\do@insidelineLhook</code>	1057, 1096, <u>1099</u>
<code>\do@insidelineRhook</code>	1097, <u>1099</u>
<code>\do@lineL</code>	<u>1013</u> , 1925, 2231
<code>\do@lineLhook</code>	1019, 1094, <u>1099</u>
<code>\do@lineR</code>	<u>1104</u> , 1936, 2273
<code>\do@lineRhook</code>	1095, <u>1099</u> , 1111
<code>\do@lockoff</code>	<u>521</u>
<code>\do@lockoffL</code>	545
<code>\do@lockoffR</code>	<u>521</u>
<code>\do@lockon</code>	<u>486</u>
<code>\do@lockonL</code>	518
<code>\do@lockonR</code>	<u>486</u>
<code>\doinsidelinehook</code>	2077, 2105
<code>\doinsidelineLhook</code>	<u>1094</u>
<code>\doinsidelineRhook</code>	<u>1094</u>
<code>\dolineLhook</code>	<u>1094</u>
<code>\dolineRhook</code>	<u>1094</u>
<code>\dolistloop</code>	742, 1615, 1635, 1642, 2350
<code>\dummy@ref</code>	559
<code>\dump@pstartL@pc</code>	<u>747</u> , 965
<code>\dump@pstartR@pc</code>	<u>747</u> , 986

E

<code>\edfont@info</code>	680, 683, 689, 692
<code>\edlabel</code>	<u>1575</u>
<code>\edtext</code>	676
<code>\eled@sectioningR@out</code>	73, 101, <u>2638</u>
<code>\eled@sections@@</code>	1039, 1071, 1926, 2232
<code>\eled@sectionsR@@</code>	70, 1131, 1937, 2274
<code>\eledpar@error</code>	<u>21</u> , 23, 26, 29, 31, 33, 35, 37, 39
<code>\eledsection@correcting@skip</code>	1091, 1901, 2173, <u>2637</u>
<code>\eledsectmark</code>	<u>14</u> , <u>2635</u>
<code>\eledsectnotoc</code>	<u>14</u> , <u>2633</u>
<code>\empty</code>	86, 89, 285, 293, 678, 687, 758, 764, 876, 926, 1255, 1337, 1345, 1443–1445, 1456, 1467, 2384, 2390, 2397, 2403
<code>\endashchar</code>	1478
<code>\endgraf</code>	960, 981, 1907, 2185
<code>\endline@num</code>	569, 575
<code>\endlock</code>	<u>667</u> , 1719, 1727, 1737
<code>\endnumbering</code>	9, <u>76</u> , 133, 827
<code>\endnumberingR</code>	48, <u>76</u> , 116, 128, 141, 827
<code>\endpage@num</code>	568, 575
<code>\endstanzaextra</code>	1739
<code>\endsub</code>	<u>634</u>
<code>\endsubline@num</code>	570, 576
<code>\enlargethispage</code>	2024
environments:	
<code>astanza</code>	11, <u>1713</u>
<code>Leftside</code>	7, <u>801</u>
<code>pages</code>	5, <u>769</u>

pairs	3, <u>769</u>
Rightside	7, <u>821</u>
\expandonce	589, 1654, 1658, 1674, 1685
\extensionchars	65, 122, 138, 146

F

\f@x@l@cks	1035
\f@x@l@cksR	1127, <u>1265</u>
\finish@Pages@notes	2328, <u>2332</u>
\finish@sameparallelpage@number	2327, <u>2612</u>
\first@linenum@out@Rfalse	<u>613</u> , 619
\first@linenum@out@Rtrue	<u>613</u>
\firstlinenum	7, <u>209</u>
\firstlinenum*	7, <u>209</u>
\firstsublinenum	7, <u>209</u>
\firstsublinenum*	7, <u>209</u>
\fix@page	346, <u>353</u>
\flag@end	<u>634</u>
\flag@start	<u>634</u>
\flush@notes	1976, 2317
\flush@notesR	<u>1465</u> , 1977, 2318
\footnotelang@lua	1668, 1679
\footnotelang@poly	1672, 1683
\footnoteXmk	7
\footnoteXnomk	7
\fullstop	248, 1475, 1477, 1479, 1481

G

\get@linelistfile	281
\get@nextboxL	2248, <u>2533</u>
\get@nextboxR	2290, <u>2533</u>
\getline@numL	1025, 1165
\getline@numR	1117, <u>1146</u>
\getlinesfrompagelistL	2208, 2310, <u>2396</u>
\getlinesfrompagelistR	2209, 2311, <u>2396</u>
\getlinesfromparlistL	2200, <u>2383</u>
\getlinesfromparlistR	2201, <u>2383</u>
\gl@p	288, 289, 296,
297, 682, 691, 721, 759, 765, 1258, 1259, 1449, 1453, 1468, 2387, 2393, 2400, 2406	
\goalfraction	6, <u>2520</u>

H

\hangingsymbol	12, 1697, 1706
\hb@xt@	1032, 1046, 1056, 1124, 1139, 1945, 2127, 2149, 2237, 2246, 2279, 2288
\hsize	895, 944, 1945, 2046–2049, 2052, 2066, 2068,
2069, 2071, 2073, 2084, 2095–2098, 2101, 2111, 2127, 2149, 2238, 2246, 2280, 2288	

I

\if@files w	1817
\if@firstcolumn	1352, 1378, 1399, 1621

\if@nobreak	855, 905
\if@noeled@sec	71, 100
\if@nopb	2009, 2024
\if@pb	2008, 2025
\if@pstarts	<u>1859</u> , 1914, 2197, 2213
\if@RTL	1084
\ifaraw@text	<u>1871</u> , 1923, 2224, 2309
\ifautopar	887, 937
\ifbool	1520, 1563
\ifboolexpr	1492, 1536, 2121, 2143
\ifbypage@	369
\ifbypage@R	<u>149</u> , 359, 1237
\ifbypstart@	598, 1966, 2548
\ifbypstart@R	<u>149</u> , 602, 1970, 2584
\ifcempty	1487, 1531
\ifcsstring	1493, 1495, 1537, 1539, 2122, 2123, 2144, 2145
\ifcsundef	586
\ifdefempty	2124, 2146
\ifdefstring	1068, 1069, 1078, 1928, 1939, 1946, 1956, 2128, 2136, 2150, 2158, 2228, 2270
\ifdim	635, 639, 641, 645, 1989, 1995, 2237, 2279, 2359, 2365, 2374, 2437, 2458
\ifdimgreater	2715, 2718
\ifdimless	2728
\iffirst@linenum@out@R	<u>613</u> , 617
\ifhbox	1948, 1953
\ifinserthangingsymbol	1693, 2501
\ifinserthangingsymbolR	1691, 1702, 2512
\ifinstanzaL	799, <u>799</u> , 1694, 2499
\ifinstanzaR	<u>799</u> , 800, 1703, 2510
\ifl@d@dash	1478
\ifl@d@elin	1480, 1481
\ifl@d@esl	1481
\ifl@d@pnum	1475, 1479
\ifl@d@ssub	1477
\ifl@dhidenum	1033, 1125
\ifl@dpagfull	2258, 2300, <u>2429</u>
\ifl@dpaging	<u>16</u>
\ifl@dpairing	<u>16</u> , 83, 1897, 2170
\ifl@dprintingpages	2627
\ifl@dsamelang	<u>1814</u>
\ifl@dsamepage	2227, 2268, <u>2429</u>
\ifl@dskipnumber	1328
\ifl@dskipversenumberR	1196, 1350
\ifl@dusedbabel	<u>1813</u>
\iflabelpstart	897, 946
\ifledgroupnotesL@	1169
\ifledgroupnotesR@	1150, 1327
\iflednopbinverse	2500, 2511
\ifledplinenum	1476

`\ifledRcol` 16,
 192, 214, 218, 222, 226, 268, 283, 347, 356, 381, 395, 412, 429, 441, 449, 470,
 515, 542, 551, 636, 642, 648, 655, 663, 668, 672, 677, 1586, 1653, 1666, 1840, 1852
`\ifluatex` 861, 911, 1058, 1074, 1667, 1678
`\ifnocritical@` 159, 167, 175, 702, 732, 2334
`\ifnofamiliar@` 698, 706, 735, 2337
`\ifnoteschanged@` 93
`\ifnumberedpar@` 870, 920, 957, 978, 1651, 1665
`\ifnumbering` 152, 866, 954
`\ifnumberingR` 46, 77, 118, 916, 975
`\ifnumberline` 1151, 1170, 1327
`\ifnumberpstart` 749, 754, 888, 938, 966, 987, 1014, 1105
`\ifnumequal` 1608
`\ifnumgreater` 1616, 1636, 1643
`\ifnumodd` 1493, 1495, 1537, 1539
`\ifodd` 1362, 1388, 1409, 1631, 2357, 2363, 2367, 2377
`\ifparledgroup` 2704, 2736
`\ifprint@last@after@pendL` 998, 2250, 2447
`\ifprint@last@after@pendR` 998, 2292, 2468
`\ifpst@rtedL` 42, 874
`\ifpst@rtedR` 42, 924
`\ifpstartnum` 1419, 1424
`\ifpstartnumR` 1374
`\ifsameparallelpagenunder` 13, 2608, 2613
`\ifshiftedpstarts` 6, 2236, 2278, 2521
`\ifsidepstartnum` 889, 939, 1376, 1397
`\ifstrempty` 899, 948, 970, 991
`\IfStrEq` 1023, 1115, 2010, 2018, 2476, 2480,
 2488, 2492, 2503, 2504, 2514, 2515, 2654, 2655, 2662, 2663, 2675, 2676, 2683
`\ifstrequal` 2707, 2713
`\ifsublines@` 246, 335, 380, 413,
 420, 451, 460, 472, 479, 491, 525, 574, 576, 1152, 1171, 1244, 1331, 1588, 1592
`\ifvbox` 1016, 1108, 1874, 1877, 2534, 2570
`\ifvoid` 802, 822, 2340, 2341
`\ifwidthliketwocolumns` 12
`\ifwrittenlinesL` 2530, 2542
`\ifwrittenlinesR` 2531, 2578
`\init@sameparallepage@number` 2193, 2605
`\init@series@eledpar` 740
`\initnumbering@sectcmd` 772, 786
`\initnumbering@sectcountR` 69, 105, 125, 2269
`\InputIfFileExists` 72
`\insert@count` 548, 1660, 1687
`\insert@countR` 549, 1656, 1676
`\insert@noterule@ledgroup` 2658, 2666, 2674
`\inserthangingsymbolfalse` 1029
`\inserthangingsymbolL` 1062, 1691
`\inserthangingsymbolR` 1691
`\inserthangingsymbolRfalse` 1121
`\inserthangingsymbolRtrue` 1119

\inserthangingsymboltrue	1027
\insertlines@listR	86, <u>256</u> , 270, 555, 1445, 1449
\inserts@list	875, 1659, 1686
\inserts@listR	925, <u>1440</u> , 1443, 1453, 1467, 1468, 1655, 1675
\istanzaLfalse	1985, 2324
\istanzaLtrue	813
\istanzaRfalse	1986, 2325
\istanzaRtrue	839
\itemcount@	1606, 1608, 1613, 1616, 1634, 1636, 1641, 1643

L

\l@d@nums	680, 683, 689, 692
\l@d@set	<u>428</u> , 663, 664
\l@dbbl@set@language	<u>1814</u> , 1837
\l@dbfnote	<u>1650</u>
\l@dc@maxchunks	881, 883, 931, 933, <u>1772</u> , 1782, 1790, 1797
\l@dcalc@maxoftwo	2202, <u>2416</u> , 2556, 2590
\l@dcalc@minoftwo	2210, 2312, <u>2416</u>
\l@dcalcnun	<u>1265</u>
\l@dchecklang	<u>1814</u>
\l@dchset@num	302, 305, <u>428</u>
\l@dcsnotetext	1615, 1618
\l@dcsnotetext@l	1618, 1635
\l@dcsnotetext@r	1618, 1642
\l@demptyd@ta	1020, 1112
\l@dend@stuff	66, 123, 139, 147
\l@dgetline@margin	190
\l@dgetsidenote@margin	1597
\l@dhidenumberfalse	1034, 1126
\l@dhidenumbertrue	1224
\l@dld@ta	1053, 1353, 1365, 1379, 1391, 1400, 1412
\l@dleftbox	<u>1000</u> , 1031, 1046, 1947, 2237, 2238, 2246
\l@dlinenumR	<u>238</u>
\l@dlsn@te	1055
\l@dmake@labelsR	<u>1575</u>
\l@dminpagelines	<u>2164</u> , 2211, 2313, 2438, 2459
\l@dnumpstartsL	749, 880, 881, 883, 885, 900, 901, 971, 972, <u>1776</u> , 1808, 1862, 1902, 1903, 1981, 2177, 2178, 2322, 2546
\l@dnumpstartsR	50, 80, 754, 930, 931, 933, 935, 949, 950, 992, 993, <u>1776</u> , 1809, 1865, 1902, 1903, 1982, 2177, 2178, 2323, 2582
\l@doldbbl@set@language	1836
\l@doldselectlanguage	1835, 1839, 1844
\l@dpagefullfalse	<u>2429</u> , 2478, 2483, 2490, 2495
\l@dpagefulltrue	<u>2429</u> , 2477, 2482, 2489, 2494
\l@dpagingfalse	771, 791
\l@dpagingtrue	785
\l@dpairingfalse	775, 790
\l@dpairingtrue	770, 784
\l@dprintingcolumnfalse	1983
\l@dprintingcolumnstrue	1900

<code>\l@printingpagesfalse</code>	2326
<code>\l@printingpagetrue</code>	2169
<code>\l@dpscl</code>	1014, 1016, 1021, 1039, 1070, 1080, 1083, 1778, 1810, 1862, 1874, 1911, 1917, 1926, 1929, 1931, 1979, 2122, 2130, 2139, 2144, 2152, 2161, 2189, 2198, 2203, 2206, 2214, 2229, 2230, 2232, 2251, 2252, 2320, 2534, 2536, 2539, 2546, 2554–2556, 2558, 2560, 2653, 2708
<code>\l@dpscR</code>	1105, 1108, 1113, 1131, 1779, 1811, 1865, 1877, 1912, 1918, 1937, 1940, 1942, 1980, 2123, 2134, 2140, 2145, 2156, 2162, 2190, 2199, 2207, 2215, 2271, 2272, 2274, 2293, 2294, 2321, 2570, 2572, 2575, 2582, 2588–2590, 2592, 2594, 2661, 2710
<code>\l@drd@ta</code>	1063, 1355, 1363, 1381, 1389, 1402, 1410
<code>\l@drightbox</code>	<u>1000</u> , 1123, 1139, 1952, 2279, 2280, 2288
<code>\l@drsn@te</code>	1064
<code>\l@dsamepagefalse</code>	<u>2429</u> , 2477, 2482, 2489, 2494
<code>\l@dsamepagetrue</code>	<u>2429</u> , 2478, 2483, 2490, 2495
<code>\l@dsetupmaxlinecounts</code>	<u>1789</u> , 1806
<code>\l@dsetuprawboxes</code>	<u>1781</u> , 1805
<code>\l@dskipnumberfalse</code>	1329
<code>\l@dskipnumbertrue</code>	1220
<code>\l@dskipversenumberfalse</code>	2249
<code>\l@dskipversenumberR</code>	<u>1195</u>
<code>\l@dskipversenumberRfalse</code>	2291
<code>\l@dskipversenumberRtrue</code>	1222
<code>\l@dunhbox@line</code>	1063, 1086, 1089
<code>\l@dusedbabelfalse</code>	<u>1813</u> , 1832
<code>\l@dusedbabeltrue</code>	<u>1813</u> , 1834
<code>\l@duselanguage</code>	<u>1824</u> , 1924, 1935, 2226, 2267
<code>\l@dzeromaxlinecounts</code>	<u>1789</u> , 1807
<code>\l@dzeropenalties</code>	886, 936, 1906, 2184
<code>\l@luatexbodydir@L</code>	864, 1077
<code>\l@luatexbodydir@R</code>	914
<code>\l@luatexpardir@L</code>	863, 1076
<code>\l@luatexpardir@R</code>	913
<code>\l@luatextextdir@L</code>	862, 1059, 1075, 1078
<code>\l@luatextextdir@R</code>	912
<code>\l@prev@nopbR</code>	57, 2640, 2647, 2648
<code>\l@prev@pbR</code>	56, 2639, 2645, 2646
<code>\l@pscl</code>	<u>1778</u>
<code>\l@pscR</code>	<u>1778</u>
<code>\labelref@list</code>	1593
<code>\labelref@listR</code>	<u>1573</u> , 1589
<code>\languagename</code>	1815, 1816, 1818–1821
<code>\last@page@num</code>	367, 373
<code>\last@page@numR</code>	<u>353</u>
<code>\lastbox</code>	1024, 1116
<code>\lastskip</code>	634, 640
<code>\Lcolwidth</code>	<u>4</u> , <u>5</u> , <u>16</u> , 787, 895, 1032, 1046, 1056, 1932, 2047, 2068, 2085, 2096, 2112, 2129, 2151, 2180
<code>\led@err@BadLeftRightPstarts</code>	<u>25</u> , 1903, 2178
<code>\led@err@Columns@InsideEnv</code>	<u>36</u> , 1898

\led@err@LeftOnRightPage	28, 2368
\led@err@Leftside@PreviousNotPrinted	32, 803
\led@err@LineationInNumbered	153
\led@err@ManyLeftnotes	1636
\led@err@ManyRightnotes	1643
\led@err@ManySidenotes	1616
\led@err@NumberingNotStarted	97
\led@err@numberingShouldHaveStarted	127
\led@err@NumberingStarted	47
\led@err@NumberingWithoutPstart	81
\led@err@Pages@InsideEnv	36, 2171
\led@err@PendNoPstart	958, 979
\led@err@PendNotNumbered	955, 976
\led@err@PstartInPstart	871, 921
\led@err@PstartNotNumbered	867, 917
\led@err@RightOnLeftPage	28, 2378
\led@err@Rightside@PreviousNotPrinted	32, 823
\led@err@TooManyPstarts	22, 882, 932
\led@mess@NotesChanged	94
\led@mess@SectionContinued	121, 137, 145
\led@nopbnumR	2644, 2645
\led@nopbR	2643, 2645
\led@pb@setting	2010, 2018, 2476, 2480, 2488, 2492, 2503, 2504, 2514, 2515
\led@pbnumR	2642, 2645
\led@pbR	2641, 2645
\led@warn@BadAction	1226
\led@warn@BadAdvancelineLine	398, 404
\led@warn@BadAdvancelineSubline	384, 390
\led@warn@BadLineation	179
\led@warn@BadSetline	653
\led@warn@BadSetlinenum	661
\led@warn@DuplicateLabel	1577
\ledgroupnotesL@false	2669
\ledgroupnotesL@true	2657
\ledgroupnotesR@false	2672
\ledgroupnotesR@true	2665
\ledllfill	1056
\lednopb	835
\lednopbnum	2503, 2641
\lednopbnumR	2514, 2641
\lednopbR	835, 2643
\ledpb	834
\ledpbnum	2504
\ledpbnumR	2515, 2641
\ledpbR	834, 2641
\ledRcol@false	1142
\ledRcol@true	1106
\ledRcolfalse	805, 841
\ledRcoltrue	825
\ledrlfill	1063, 2056, 2078

<code>\ledsavedprintlines</code>	10, 1473
<code>\ledsectnomark</code>	1069
<code>\ledsectnotoc</code>	1068, 2228, 2270
<code>\ledstrutL</code>	2238, 2246, 2352
<code>\ledstrutR</code>	2280, 2288, 2352
<code>\ledthegoal</code>	2437, 2458, 2520
<code>\leftlinenumR</code>	238 , 1353, 1365
<code>\leftpstartnumL</code>	1374
<code>\leftpstartnumR</code>	1374
<code>Leftside (environment)</code>	7, 801
<code>\Leftsidehook</code>	811, 816
<code>\Leftsidehookend</code>	815, 816
<code>\letcs</code>	587, 588, 714, 720, 723, 1014, 1105
<code>\line@list</code>	687, 691
<code>\line@list@stuff</code>	138
<code>\line@list@stuffR</code>	65, 122, 146, 615
<code>\line@list@version</code>	621
<code>\line@listR</code>	89, 256 , 269, 576, 678, 682
<code>\line@margin</code>	195, 1384
<code>\line@marginR</code>	188 , 1358, 1405
<code>\line@num</code>	370, 402, 403, 405, 423, 434, 435, 463, 598, 1177, 1591, 2549
<code>\line@numR</code>	58, 245, 252 , 307, 341, 360, 396, 397, 399, 416, 430, 431, 454, 569, 573, 602, 1158, 1238, 1247, 1336, 1338, 1340, 1341, 1587, 2585
<code>\lineation</code>	184, 185, 836
<code>\lineation*</code>	8, 184
<code>\lineationR</code>	8, 151 , 186, 836
<code>\linenum@out</code>	631, 637, 643, 649, 656, 664, 669, 673, 1885, 1967, 2410
<code>\linenum@outR</code>	612 , 618, 620, 621, 626, 627, 633, 636, 642, 648, 655, 663, 668, 672, 1890, 1971, 2413, 2641–2644
<code>\linenumberlist</code>	1337, 1341
<code>\linenumincrement</code>	7, 209
<code>\linenumincrement*</code>	7, 209
<code>\linenummargin</code>	188
<code>\linenumr@p</code>	1476, 1480, 1587, 1591
<code>\linenumrepR</code>	235 , 245
<code>\linenumsep</code>	240, 242, 1421, 1424, 1433, 1436
<code>\linesinpar@listL</code>	263 , 277, 599, 2384, 2387
<code>\linesinpar@listR</code>	263 , 273, 603, 2390, 2393
<code>\lineskip</code>	1525, 1568
<code>\linesonpage@listL</code>	278, 607, 2397, 2400
<code>\linesonpage@listR</code>	274, 610, 2403, 2406
<code>\list@clear</code>	269–274, 277, 278, 280, 875, 925
<code>\list@clearing@reg</code>	276
<code>\list@create</code>	256–259, 263–265, 577, 585, 709, 745, 746, 1440, 1573
<code>\list@pstartL@pc</code>	745 , 748, 758, 759
<code>\list@pstartR@pc</code>	745 , 753, 764, 765
<code>\listbreak</code>	2343, 2347
<code>\listxadd</code>	375, 2645–2648
<code>\lock@disp</code>	1297, 1301, 1306
<code>\lock@off</code>	512, 513, 521 , 672, 673

\lockon 668, 669

M

\managestanza@modulo 1748
 \maxchunks 3, 1772
 \maxlinesinpar@list 263, 280
 \memorydump 9, 810, 830
 \memorydumpL 132, 810
 \memorydumpR 132, 830
 \message 64
 \multiply 1271, 2050, 2099

N

\n@num 549
 \namebox 1016, 1021, 1108, 1113, 1756, 1874, 1877, 2534, 2570
 \NeedsTeXFormat 2
 \new@line 1086, 1089
 \new@lineL 630, 1061
 \new@lineR 632
 \newbool 700, 703
 \newbox 846, 1000, 1001, 1757
 \newcommandx 854, 904, 953, 974, 1736
 \newcounter 105–108, 200, 202, 204, 206, 849, 851, 2605
 \newhookcommand@series 733, 736
 \newif 6, 13, 43, 149, 150, 613, 799, 800,
 998, 999, 1196, 1428, 1691, 1813, 1859, 1871, 2008, 2009, 2429, 2431, 2530, 2531
 \newlength 2036, 2039
 \newmarks 2649–2651
 \newnamebox 1756, 1784, 1785, 2335, 2338
 \newnamecount 1767, 1792
 \newsavebox 1894, 1895
 \newseries@eledpar 697, 741
 \newwrite 612, 2638
 \next@absline 2011, 2013, 2015, 2481–2483
 \next@abslineR 2012, 2014, 2016, 2493–2495
 \next@action 297
 \next@actionline 294, 296
 \next@actionlineR 286, 288, 1189, 1234, 1256, 1258
 \next@actionR 289, 1190, 1235, 1236, 1241, 1242, 1250, 1259
 \next@insert 876
 \next@insertR 926, 1444, 1447, 1449, 1452, 1456
 \next@page@num 374, 446
 \next@page@numR 62, 310, 312, 364, 443
 \noindent 901, 950, 972, 993
 \normal@page@break 375
 \normal@page@breakR 55
 \normal@pars 79, 879, 929
 \normalbfnoteX 1664
 \notefontsetup 2695
 \noteschanged@true 87, 90, 679, 688, 1446

<code>\notesXwidthliketwocolumns</code>	5
<code>\num@lines</code>	960, 1907, 2185
<code>\num@linesR</code>	845, 981, 1908, 2186
<code>\numberedpar@true</code>	896, 945
<code>\numberingRfalse</code>	78
<code>\numberingRtrue</code>	52, 116, 142
<code>\numberingtrue</code>	134
<code>\numberpstartfalse</code>	10
<code>\numberpstarttrue</code>	10
<code>\numdef</code>	715, 1070, 1634, 1641, 2011, 2012, 2481, 2493, 2708, 2710
<code>\numgdef</code>	1606, 1613, 2502, 2513
<code>\numlabfont</code>	245
<code>\numpagelinesL</code>	2164, 2235, 2259, 2263, 2438, 2504, 2728
<code>\numpagelinesR</code>	2164, 2277, 2301, 2305, 2459, 2515

O

<code>\old@footnote</code>	723, 727
<code>\old@otherlanguage</code>	1849
<code>\old@startstanza</code>	812, 813, 838, 839
<code>\oldchapter</code>	782, 792
<code>\one@line</code>	1021, 1024, 1063, 1086, 1089
<code>\one@lineR</code>	845, 1113, 1116
<code>\onlysideX</code>	6
<code>\onlyXside</code>	6
<code>\openout</code>	73, 620, 627
<code>\otherlanguage</code>	1849, 1850

P

<code>\p@pstartL</code>	898
<code>\p@pstartR</code>	947
<code>\PackageError</code>	21
<code>\page@action</code>	311, 440, 561
<code>\page@num</code>	292, 372, 1386
<code>\page@numR</code>	261, 284, 362, 568, 573, 1236, 1360, 1407, 1629
<code>\pagebreak</code>	2025
<code>\pagegoal</code>	2528
<code>\Pages</code>	5, 33, 35, 37, 2168
<code>pages (environment)</code>	5, 769
<code>\pagetotal</code>	2240, 2242, 2282, 2284, 2359, 2365, 2374, 2437, 2458
<code>pairs (environment)</code>	3, 769
<code>\paperwidth</code>	1085, 1088
<code>\par@line</code>	961, 1909, 2187
<code>\par@lineR</code>	845, 982, 1910, 2188
<code>\parbox</code>	1932, 1943, 2129, 2133, 2151, 2155
<code>\pardir</code>	863, 913, 1076
<code>\parledgroup@</code>	1023, 1115, 2649, 2675
<code>\parledgroup@beforenotes@save</code>	969, 990, 2735
<code>\parledgroup@beforenotesL</code>	2733
<code>\parledgroup@beforenotesR</code>	2733
<code>\parledgroup@correction@notespacing</code>	2245, 2287, 2722

<code>\parledgroup@correction@notespacing@final</code>	2564, 2598, 2703
<code>\parledgroup@correction@notespacing@init</code>	2264, 2306, 2698, 2706
<code>\parledgroup@notes@endL</code>	2537, 2540, 2563, 2668
<code>\parledgroup@notes@endR</code>	2573, 2576, 2597, 2671
<code>\parledgroup@notes@startL</code>	1023, 2652, 2668
<code>\parledgroup@notes@startR</code>	1115, 2652, 2668
<code>\parledgroup@notespacing@correction</code>	2693, 2725–2727
<code>\parledgroup@notespacing@correction@accumulated</code>	2699, 2705, 2726
<code>\parledgroup@notespacing@correction@modulo</code>	2700, 2727–2729
<code>\parledgroup@notespacing@set@correction</code>	2174, 2693
<code>\parledgroup@series</code>	2650, 2654, 2655, 2662, 2663, 2678, 2679, 2685, 2686
<code>\parledgroup@type</code>	2651, 2654, 2655, 2662, 2663, 2676, 2683
<code>\parledgroupnotespacing</code>	2692, 2695
<code>\parledgroupseries@</code>	2649
<code>\parledgrouptrue</code>	11
<code>\parledgrouptype@</code>	2649
<code>\patchcmd</code>	2609
<code>\pausenumbering</code>	828
<code>\pausenumberingR</code>	115, 828
<code>\pend</code>	8, 809, 833, 872, 1738
<code>\pendL</code>	809, 953
<code>\pendR</code>	833, 922, 974
<code>\pretocmd</code>	2626
<code>\prev@abslineverse</code>	2502–2504, 2513–2515
<code>\prev@nopbR</code>	2639
<code>\prev@pbR</code>	2639
<code>\prevgraf</code>	960, 981, 1907, 1908, 2185, 2186
<code>\print@columnseparator</code>	1951, 1988, 2132, 2154
<code>\print@eledsectionL</code>	1066, 1932, 2233
<code>\print@eledsectionR</code>	1146, 1943, 2275
<code>\print@last@after@pendLfalse</code>	2450
<code>\print@last@after@pendRfalse</code>	2471
<code>\print@last@after@pendRtrue</code>	2600
<code>\print@lineL</code>	1041, 1051
<code>\print@lineR</code>	1133, 1146
<code>\print@notesX</code>	2195
<code>\print@notesX@forpages</code>	1486, 2195
<code>\print@Xnotes</code>	2194
<code>\print@Xnotes@forpages</code>	1486, 2194
<code>\printlines</code>	1484
<code>\printlinesR</code>	10, 1473
<code>\ProcessOptions</code>	15
<code>\protected@csxdef</code>	725
<code>\protected@edef</code>	897, 946
<code>\protected@write</code>	1818
<code>\protected@xdef</code>	1673, 1684
<code>\providebool</code>	578
<code>\ProvidesPackage</code>	3
<code>\pst@rtedLfalse</code>	42
<code>\pst@rtedLtrue</code>	135, 877

`\pst@rtedRfalse` 51, 84
`\pst@rtedRtrue` 119, 143, 927
`\pstart` 8, 23, 27, 807, 832, 1732
`\pstartinfootnote` 160, 168, 176
`\pstartL` 807, 848
`\pstartnumfalse` 1421, 1426
`\pstartnumRfalse` 1433, 1438
`\pstartnumRtrue` 1429, 1916, 2595
`\pstartnumtrue` 1915, 2561
`\pstartR` 832, 848

R

`\Rcolwidth` 4,
 5, 16, 788, 944, 1124, 1139, 1943, 2048, 2069, 2086, 2097, 2113, 2133, 2155, 2181
`\read@linelist` 267, 616
`\rem@inder` 1341, 1343–1345
`\RequirePackage` 5
`\resetprevline@` 1968, 1972, 2550, 2586
`\restore@pstartL@pc` 757, 1919, 2216, 2562
`\restore@pstartR@pc` 757, 1920, 2217, 2596
`\resumenumbering` 829
`\resumenumberingR` 115, 829
`\rightlinenumR` 238, 1355, 1363
`\rightpstartnumL` 1374
`\rightpstartnumR` 1374
`Rightside (environment)` 7, 821
`\Rightsidehook` 816, 837
`\Rightsidehookend` 816, 842
`\rlap` 1355, 1363, 1381, 1389, 1402, 1410
`\Rlineflag` 10, 233, 245, 1476, 1480, 1579
`\rule` 2028

S

`\sameparallelpagenumbertrue` 6, 14
`\savebox` 1932, 1943
`\sc@n@list` 1342, 1344
`\secdef` 797
`\section@num` 136–138
`\section@numR` 40, 53, 64, 65, 72, 73, 120–122, 144–146
`\select@language` 1816, 1818–1821, 1855
`\selectlanguage` 1824
`\set@line` 676
`\set@line@action` 304, 409, 418, 425, 448, 563
`\setl@dlp@rbox` 1622, 1637, 1639
`\setl@drp@rbox` 1624, 1632, 1644
`\setline` 651
`\setlinenum` 659
`\setnamebox` 885, 935, 1756
`\setnotepositionliketwocolumns@C` 2042
`\setnotepositionliketwocolumns@L` 2042

<code>\setnotepositionliketwocolumns@R</code>	2042
<code>\setnotespositionliketwocolumns@C</code>	2081
<code>\setnotespositionliketwocolumns@L</code>	2059
<code>\setnotespositionliketwocolumns@R</code>	2108
<code>\setpositionliketwocolumns@C</code>	2042, 2076
<code>\setpositionliketwocolumns@L</code>	2042, 2055
<code>\setpositionliketwocolumns@R</code>	2042, 2104
<code>\setprintlines</code>	1474
<code>\setwidthliketwocolumns@C</code>	2042, 2063
<code>\setwidthliketwocolumns@L</code>	2042, 2042
<code>\setwidthliketwocolumns@R</code>	2042, 2093
<code>\shiftedpstartsfalse</code>	8
<code>\shiftedpstartstrue</code>	7, 9, 10
<code>\shiftedversesfalse</code>	8
<code>\shiftedversestrue</code>	7
<code>\sidenote@margin</code>	1599, 1602
<code>\sidenote@marginR</code>	1596, 1627
<code>\sidenotecontent@</code>	1605, 1610, 1611, 1622, 1624, 1632, 1633, 1637, 1639, 1640, 1644
<code>\sidenotemargin</code>	1596
<code>\sidenotemargin*</code>	1596
<code>\sidenotesep</code>	1611
<code>\skip</code>	1502, 1510, 1546, 1554, 2678, 2685
<code>\skip@lockoff</code>	513, 521
<code>\skipnumbering</code>	11, 676
<code>\smash</code>	2028
<code>\splitbotmarks</code>	2675, 2676, 2678, 2679, 2683, 2685, 2686
<code>\splitfirstmarks</code>	1023, 1115, 2654, 2655, 2662, 2663
<code>\splittopskip</code>	1018, 1110, 1522, 1565
<code>\stanza@count</code>	1716, 1729, 1743
<code>\stanza@hang</code>	1718, 1751
<code>\stanza@modulo</code>	1716, 1746
<code>\stanzaindentbase</code>	1696, 1705, 1744, 1747
<code>\startlock</code>	667
<code>\startstanzahook</code>	1714
<code>\startsub</code>	634
<code>\stepcounter</code>	2376
<code>\sub@action</code>	320, 469, 562
<code>\sub@change</code>	63, 314, 315, 321
<code>\sub@lock</code>	1172
<code>\sub@lockR</code>	60, 329, 331, 333, 336, 487, 493, 494, 496, 497, 527, 528, 530, 1153, 1212, 1214, 1215, 1217, 1278, 1318, 1320, 1322
<code>\sub@off</code>	642, 643
<code>\sub@on</code>	636, 637
<code>\subline@num</code>	247, 370, 388, 389, 391, 421, 461, 1173, 1178, 1592
<code>\subline@numR</code>	248, 252, 337, 341, 360, 382, 383, 385, 414, 452, 570, 574, 1154, 1159, 1238, 1245, 1332, 1333, 1588
<code>\sublinenumincrement</code>	7, 209
<code>\sublinenumincrement*</code>	7, 209
<code>\sublinenumr@p</code>	1477, 1481, 1588, 1592
<code>\sublinenumrepR</code>	235, 248

<code>\sublines@false</code>	61, 318, 1202
<code>\sublines@true</code>	316, 1200
<code>\subblock@disp</code>	1280, 1284, 1289
<code>\symplinenum</code>	1476
<code>\sza@penalty</code>	1724, 1728

T

<code>\temp</code>	2045, 2046, 2049–2052, 2065, 2066, 2071–2073, 2082, 2084, 2087–2090, 2094, 2095, 2098–2101, 2109, 2111, 2114–2117
<code>\temp@</code>	1070, 1071
<code>\temp@spacing</code>	2695, 2696
<code>\tempa</code>	2083, 2085–2087, 2110, 2112–2114
<code>\textdir</code>	862, 912, 1059, 1075
<code>\textwidth</code>	17, 19, 787, 788, 2180, 2181
<code>\theledlanguageL</code>	1824, 1924, 2226
<code>\theledlanguageR</code>	1824, 1935, 2267
<code>\thepage</code>	631, 633, 2609
<code>\thepstart</code>	808, 831
<code>\thepstartL</code>	10, 808, 850, 891, 898, 1420, 1425
<code>\thepstartR</code>	10, 831, 852, 940, 947, 1432, 1437
<code>\this@line@list@version</code>	621
<code>\thisfootnote</code>	1673, 1674, 1684, 1685
<code>\thr@@</code>	496, 505, 528, 535, 1207, 1215
<code>\togglefalse</code>	713
<code>\toggletrue</code>	711
<code>\topskip</code>	2359

U

<code>\unexpanded</code>	996
<code>\unhbox</code>	1761, 1947, 1952, 2238, 2246, 2280, 2288
<code>\unhnamebox</code>	1756
<code>\unless</code>	159, 167, 175, 698, 702, 706, 732, 735
<code>\unvbox</code>	1024, 1116, 1526, 1569, 1763
<code>\unvnamebox</code>	1756
<code>\usebox</code>	1949, 1954
<code>\usenamecount</code>	1717, 1723, 1767, 1799, 2203, 2536, 2539, 2556, 2558, 2572, 2575, 2590, 2592, 2653, 2661

V

<code>\value</code>	1742
<code>\vbadness</code>	1017, 1109
<code>\vbfnoteX</code>	1674, 1685
<code>\vbox</code>	885, 935, 1523, 1566, 1932, 1943, 2621, 2628
<code>\vl@dbfnote</code>	1654, 1658
<code>\vsplit</code>	1021, 1113, 1524, 1567

W

<code>\widthliketwocolumns</code>	5
<code>\widthliketwocolumnstrue</code>	12
<code>\WithSuffix</code>	184, 229–232, 1596

<code>\writtenlinesLfalse</code>	2191, 2547
<code>\writtenlinesLtrue</code>	2544
<code>\writtenlinesRfalse</code>	2192, 2583
<code>\writtenlinesRtrue</code>	2580

X

<code>\xifinlist</code>	1039, 1071, 1131, 1926, 1937, 2232, 2274
<code>\xifinlistcs</code>	2013–2016, 2019–2022, 2477, 2478, 2482, 2483, 2489, 2490, 2494, 2495
<code>\Xnoteswidthliketwocolumns</code>	5
<code>\xpg@main@language</code>	1856, 1857
<code>\xright@appenditem</code>	442, 443, 445, 446, 450, 457, 459, 466, 471, 473, 475, 478, 480, 482, 490, 492, 501, 524, 526, 533, 555, 572, 589, 599, 603, 607, 610, 716, 748, 753, 1587, 1591, 1654, 1658, 1674, 1685
<code>\xspace</code>	728

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	69
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	72
\Pages: Added \l@duselanguage to \Pages	81
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	48
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match neweledmac	52
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	50
\do@lineL: Added \do@lineLhook to \do@lineL	46
Simplified \do@lineL by using macros for some common code	46
\do@lineR: Changed \do@lineR similarly to \do@lineL	48
Leftside: Added hooks into Leftside environment	40
\flag@end: Removed extraneous spaces from \flag@end	34
\ifledRcol: Moved \ifl@dpairing toeledmac	16
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac	17
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	23
\l@dnumprintstartsR: Moved \l@dnumprintstartsL toeledmac	68
\ledsavedprintlines: Simplified \printlinesR by using \setprintlines	58
\ledstrutR: Added \ledstrutL and \ledstrutR	84
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX	64
\Pages: Added \ledstrutL to \Pages	81
Added \ledstrutR to \Pages	82
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	40
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	22
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Don't just set \line@marginR in \linenummargin	21
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	83
v0.3.c.	
General: Compatibilty with Polyglossia	1
v0.4.0.	
General: No more ledparpatch. All patches are now in the main file.	1
v0.5.0.	
General: Corrections about \section and other titles in numbered sections	1
v0.6.0.	
General: Be able to use \chapter in parallel pages.	1
v0.7.0.	
General: Option 'shiftedverses' which make there is no blank between two parallel verses with unequal length.	1

v0.8.0.	
General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande <code>\hangingsymbol</code> to define the character.	1
v0.9.0.	
General: Possibility to number <code>\pstart</code> .	10
Possibility to number the pstart with the commands <code>\numberpstarttrue</code> .	1
<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code>	16
v0.9.1.	
General: The numbering of the pstarts restarts on each <code>\beginnumbering</code> .	1
v0.9.2.	
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
v0.9.3.	
General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with <code>memoir</code> class.	1
v0.10.0.	
General: <code>\edlabel</code> commands on the right side are now correctly indicated.	1
<code>\edlabel</code> commands which start a paragraph are now put in the right place.	1
v0.11.0.	
General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by pstart (like in <code>eledmac 0.15</code>).	46
Lineation can be by pstart (like in <code>eledmac 0.15</code>).	20
New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	65
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code>).	52
<code>\Columns</code> : Line numbering by pstart.	73
<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>).	89
Pstart number can be printed in side	90
<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	65
v0.12.0.	
General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	65
v1.0.0.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code> .	1
Debug in lineation by pstart	20
v1.0.1.	
General: Correction on <code>\numberonlyfirstinline</code> with lineation by pstart or by page.	1
v1.1.0.	
General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code> .	1
<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from <code>eledmac</code>).	42
v1.1.1.	
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1.	42
v1.1.2.	
<code>\affixside@noteR</code> : Remove spurious space between line number and line content	63
v1.2.0.	
General: Support for <code>\led<section></code> commands in parallel texts.	1

v1.2.1.	
<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once.	19
v1.3.0.	
<code>\edtext</code> : Manage RTL language.	35
v1.3.1.	
<code>\l@dbfnote</code> : Compatibility of standard footnotes with eledmac when theses footnotes contain any commands.	64
v1.3.2.	
General: Debug with some classes.	1
v1.3.3.	
General: Debugging the left notes of the right column.	63
<code>\l@dbfnote</code> : Spurious space with footnote in right column.	64
v1.3.4.	
General: Allow use of commands in sidenotes, as introduced by eledmac 1.0. . .	63
v1.3.5.	
<code>\normalbfnoteX</code> : Allows one to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded.	64
v1.4.0.	
General: Added <code>\do@insidelineLhook</code> and <code>\do@insidelineRhook</code>	48
v1.4.1.	
<code>\normalbfnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text.	64
<code>astanza</code> : Enable the use of <code>stanzaindentsrepetition</code> within <code>astanza</code> environment.	66
v1.4.3.	
General: Corrects a false hanging verse when a verse is exactly the length of a line.	1
<code>\inserthangingsymbolR</code> : Hanging verse is no longer automatically flush right.	65
<code>\pendL</code> : Spurious spaces in <code>\pendL</code>	44
<code>\pendR</code> : Spurious spaces in <code>\pstartR</code>	45
<code>\pstartR</code> : Spurious spaces in <code>\pstartL</code> and <code>\pstartR</code>	42
v1.5.0.	
General: Add, as in eledmac, features to manage page breaks.	1
<code>\sublinenumincrement*</code> : Add starred version of <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\firstsublinenum</code> , <code>\sublinenumincrement</code> to change both Left and Right-side.	22
v1.6.0.	
General: Add tool and documentation for parallel ledgroups	13
v1.7.0.	
General: Add, as in eledmac, features to make crossrefs with <code>pstart</code> numbers. . .	1
v1.8.0.	
General: <code>\beginnumbering</code> is defined only on eledmac, not on eledpar.	17
<code>\l@dlsnote</code> , <code>\l@drsnote</code> and <code>\l@dcsnote</code> defined only one time, in eledmac. .	63
Add <code>\beforecolumnseparator</code> and <code>\aftercolumnseparator</code>	4
Add <code>\columnspanposition</code>	4
Add, as in eledmac, new system of sectioning commands.	1
Add, as in eledmac, option to insert something after <code>\pends</code> / verses.	1
Add, as in eledmac, option to insert something between <code>\pstarts</code> / verse. . . .	1
Change <code>\do@lineR</code> and <code>\do@lineR</code> to allow new sectioning commands.	46
Compatibility with <code>musixtex</code>	1
Debug eledmac sectioning command after using <code>\resumenumbering</code>	1

New sectioning commands, as in <code>eledmac</code> .	14
<code>\Columns</code> : Modify <code>\Columns</code> to enable to add section's title.	72
Suppress <code>\l@dcchecklang</code> from <code>\Columns</code> .	72
<code>\l@dcchecklang</code> : Suppress <code>\l@dcchecklang</code> which didn't work and was not logical, because both columns could have the same language but not the main language of the document.	69
<code>\Pages</code> : Modify <code>\Pages</code> to enable to add section's title.	79
<code>\pendL</code> : As in <code>eledmac</code> , <code>\pendL</code> can have an optional argument.	44
<code>\pendR</code> : As in <code>eledmac</code> , <code>\pendR</code> can have an optional argument.	45
<code>\print@columnseparator</code> : Move some code of <code>\Columns</code> to <code>\print@columnseparator</code> .	74
<code>\pstartR</code> : As in <code>eledmac</code> , <code>\pendL</code> and <code>\pendR</code> can have an optional argument.	42
<code>\sidenotemargin*</code> : <code>\sidenotemargin</code> is now directly defined in <code>eledmac</code> to be able to manage <code>eledpar</code> .	63
Add <code>\sidenotemargin*</code>	63
<code>\theledlanguageR</code> : Correct left/right language setting with <code>polyglossia</code> .	70
v1.8.1.	
<code>\do@lineL</code> : Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?).	46
<code>\do@lineR</code> : Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?).	48
v1.8.2.	
General: Debug <code>\eledxxx</code> with some paper sizes	1
Debug left and side note (bugs added by 1.8.0)	1
<code>\eledpar@error</code> : Errors specific to <code>eledpar</code> send to <code>eledpar</code> handbook	16
<code>\flag@end</code> : <code>\flag@start</code> and <code>\flag@end</code> are now defined only one time for <code>eledmac</code> and <code>eledpar</code>	34
<code>\lineation*</code> : Add <code>\lineation*</code>	21
v1.8.3.	
General: Add <code>\noeledxxx</code> , as in <code>eledmac</code>	1
<code>\doinsidelineRhook</code> : Added <code>\dolineLhook</code> , <code>\dolineRhook</code> , <code>\doinsidelineLhook</code> and <code>\doinsidelineRhook</code>	48
<code>\Pages</code> : Debug blank pages when using optional argument in the last <code>\pend</code> .	79
<code>\resumenumberingR</code> : Debug <code>\resumenumberingR</code>	19
v1.9.0.	
General: Add <code>\AtBeginPairs</code> macro.	4
Compatibility with <code>\Xnoteswidthliketwocolumns</code> and <code>\notesXwidthliketwocolumns</code>	1
<code>\ifwidthliketwocolumns</code> : Added <code>widthliketwocolumns</code> option	15
<code>\theledlanguageR</code> : Debug left/right language switching with <code>polyglossia</code> . Don't write in <code>.aux</code> file when setting left/right lines.	70
v1.9.1.	
<code>\ifledRcol</code> : Moved <code>\ifl@dpadding</code> to <code>eledmac</code>	16
v1.10.0.	
General: Compatibility with <code>\AtEveryPstart</code> and <code>\AtEveryPend</code>	1
Restore critical notes in <code>\eledsection</code> in parallel columns (this bug was added in 1.8.2).	1
<code>\Pages</code> : Debug wrong pages splitting when no optional argument is used in last <code>\pend</code> (bug was added in v1.8.3).	79

Debug wrong parallel pages synchronization when an <code>\edtext</code> falls accross two pages.	79
v1.10.1.	
<code>\line@list@stuffR</code> : Revert modification of 1.4.2, which makes bugs with numbering. Leave vertical mode to solve spurious space before minipage.	33
v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles.	1
<code>\edtext</code> : <code>\critext</code> and <code>\edtext</code> are now defined only in <code>eledmac</code>	35
v1.12.0.	
General: Compatibility with Lua ^A T _E X RTL languages.	1
<code>\Columns</code> : Add <code>\l@dprintingcolumnstrue</code>	72
<code>\edlabel</code> : <code>\edlabel</code> and <code>\edindex</code> works now with <code>hyperref</code> when using <code>eledpar</code>	62
<code>\edlabel</code> is now defined only one time for both <code>eledmac</code> and <code>eledpar</code>	62
<code>\Pages</code> : Add <code>\l@dprintingpagestrue</code>	79
<code>\print@eledsectionL</code> : Compatibility with Lua ^A T _E X RTL languages.	47
<code>\print@eledsectionR</code> : Compatibility with Lua ^A T _E X RTL languages.	49
<code>\print@lineL</code> : Compatibility with Lua ^A T _E X RTL languages.	47
v1.12.1.	
<code>\print@eledsectionL</code> : Fixes bug with Lua ^A T _E X RTL <code>\eledsection</code>	47
v1.13.0.	
General: Fix bug in <code>shiftedpstarts</code> when size difference between <code>pstarts</code> is very important.	1
With parallel pages, long notes can now flow from the Left to the right side and from the Right to the left side.	1
<code>\clearl@drightpage</code> : Use <code>\newpage</code> instead of <code>\clearpage</code>	85
<code>\ifledRcol</code> : Remove false boolean settings which are not needed.	16
<code>\Pages</code> : Prevent false overfull hboxes when using <code>\Pages</code> outside of pages environment.	79
When using <code>shiftedpstarts</code> option, a <code>\l@dleftbox</code> with a null height will advance the <code>\pagetotal</code> in any case.	79
<code>astanza</code> : Enable the use of optional argument of <code>&</code> in <code>astanza</code> environment. ..	66
v1.13.1.	
<code>\correct@footinsX@box</code> : Call <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> directly in <code>\print@notesX@forpages</code> and <code>\print@Xnotes@forpages</code>	58
Correct <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code>	58
<code>\Pages</code> : Prevent false empty page after <code>\Pages</code> (bug added in 1.13.0)	79
v1.14.0.	
General: Fix bug with line number position when using <code>\eledsection</code> and similar commands for RTL texts with Lua ^A T _E X.	1
The <code>\newifs</code> are not followed by boolean values set to false, because it is the T _E X default setting.	1
v1.15.0.	
General: Add <code>\AtEveryPstartCall</code>	1
Add <code>sameparallelpagenumber</code> option.	5
Fix vertical spurious space before right <code>\eledchapter</code> (bug added in v1.13.0).	1
Prevent vertical space when using <code>\AtEveryPstart</code> or <code>\AtEveryPend</code> with a command which prints nothing	1
<code>\do@actions@nextR</code> : Add action 1008 and 1009	50
<code>\inserthangingsymbolR</code> : Prevent more efficiently the column separator from shifting when a verse is hanging	65

<code>\lineationR</code> : As <code>\lineation</code> , <code>\lineationR</code> automatically set the <code>\pstartinfootnote</code> .	
.....	20
<code>\n@num</code> : <code>\n@num</code> defined only one time for both <code>eledmac</code> and <code>eledpar</code> .	31
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>eledmac</code> and <code>eledpar</code>	35
v1.16.0.	
General: Error message when calling <code>\Pages</code> inside ‘pages’ environment and <code>\Columns</code> inside ‘pairs’ environment.	1
Error message when starting a Leftside/a Rightside while the previous one has not been yet typeset.	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code> .	1
.....	1
Fix bug with <code>nofamiliar / nocritical</code> option of <code>eledmac</code> .	1
New package option <code>sameparallelpagenum</code> to have the same page number for both left and right side.	1
<code>\newseries@eledpar</code> : Fix bug with <code>\onlysideX</code> .	36
v1.16.1.	
General: Write information about line-list file version in the correct file.	1
v1.16.2.	
General: Fix bug when adding empty lines before a <code>\pend</code> in combination with some specific penalties setting.	1
v1.17.0.	
General: Add compatibility of optional argument of <code>\pstart/\pend</code> and <code>\AtEveryPstart/\AtEveryPend</code> with two columns mode.	1
v1.17.0a.	
General: <code>Eledpar</code> support ends. Migrate to <code>reledpar</code> .	1
v1.17.1.	
General: Changes some internal code in order to provide compatibility with \LaTeX release of october 2015	1