

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

Peter Wilson  
Herries Press<sup>†</sup>  
Maïeul Rouquette<sup>‡</sup>

## Abstract

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

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

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

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

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

## Contents

<b>1 Introduction</b>	<b>3</b>
<b>2 The <b>eledpar</b> package</b>	<b>4</b>
2.1 General . . . . .	4
<b>3 Parallel columns</b>	<b>5</b>
<b>4 Facing pages</b>	<b>6</b>

---

\*This file (**eledpar.dtx**) has version number v1.10.0, last revised 2015/01/12.

<sup>†</sup>herries dot press at earthlink dot net

<sup>‡</sup>maieul at maieul dot net

<b>5 Left and right texts</b>	<b>7</b>
<b>6 Numbering text lines and paragraphs</b>	<b>9</b>
<b>7 Verse</b>	<b>10</b>
<b>8 Side notes</b>	<b>12</b>
<b>9 Parallel ledgroups</b>	<b>12</b>
9.1 Parallel ledgroups and <code>setspace</code> package . . . . .	14
<b>10 Sectioning commands</b>	<b>14</b>
<b>11 Implementation overview</b>	<b>14</b>
<b>12 Preliminaries</b>	<b>14</b>
12.1 Messages . . . . .	15
<b>13 Sectioning commands</b>	<b>16</b>
<b>14 Line counting</b>	<b>19</b>
14.1 Choosing the system of lineation . . . . .	19
14.2 Line-number counters and lists . . . . .	22
14.3 Reading the line-list file . . . . .	23
14.4 Commands within the line-list file . . . . .	24
14.5 Writing to the line-list file . . . . .	32
<b>15 Marking text for notes</b>	<b>34</b>
<b>16 Parallel environments</b>	<b>36</b>
<b>17 Paragraph decomposition and reassembly</b>	<b>38</b>
17.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	38
17.2 Processing one line . . . . .	42
17.3 Line and page number computation . . . . .	45
17.4 Line number printing . . . . .	48
17.5 Pstart number printing in side . . . . .	50
17.6 Add insertions to the vertical list . . . . .	52
17.7 Penalties . . . . .	52
17.8 Printing leftover notes . . . . .	53
<b>18 Footnotes</b>	<b>54</b>
18.1 Normal footnote formatting . . . . .	54
<b>19 Cross referencing</b>	<b>54</b>
<b>20 Side notes</b>	<b>56</b>

<i>List of Figures</i>	3
<b>21 Familiar footnotes</b>	<b>57</b>
<b>22 Verse</b>	<b>58</b>
<b>23 Naming macros</b>	<b>60</b>
<b>24 Counts and boxes for parallel texts</b>	<b>61</b>
<b>25 Fixing babel</b>	<b>62</b>
<b>26 Parallel columns</b>	<b>65</b>
<b>27 Parallel pages</b>	<b>71</b>
<b>28 Sections' titles' commands</b>	<b>82</b>
<b>29 Page break/no page break, depending on the specific line</b>	<b>82</b>
<b>30 Parallel ledgroup</b>	<b>83</b>
<b>31 The End</b>	<b>86</b>
<b>Appendix A Some things to do when changing version</b>	<b>87</b>
Appendix A.1 Migration to eledpar 1.4.3 . . . . .	87
<b>References</b>	<b>87</b>
<b>Index</b>	<b>87</b>
<b>Change History</b>	<b>98</b>

## List of Figures

## 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<sup>A</sup>T<sub>E</sub>X. 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  $\TeX$  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 the box as though it was being typeset as normal onto a page and any notes are stored without being typeset. Then each typeset line is extracted from the box and any notes for that line are recalled. The line, with any notes, is then output for printing, possibly with a line number attached. Effectively, all the text is typeset and then afterwards all the notes are typeset.

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

both the number of potential boxes and memory are limited. If TeX's memory is overfilled the recourse is to reduce the amount of text stored before printing.

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

```
\maxchunks{5120}
```

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

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

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

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

### 3 Parallel columns

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

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

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

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

`\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.
<code>\Lcolwidth</code>	The lengths <code>\Lcolwidth</code> and <code>\Rcolwidth</code> are the widths of the left and right columns, respectively. By default, these are: <code>\setlength{\Lcolwidth}{0.45\textwidth}</code> <code>\setlength{\Rcolwidth}{0.45\textwidth}</code>
<code>\Rcolwidth</code>	
	They may be adjusted if one text tends to be ‘bulkier’ than the other.
<code>\columnrulewidth</code>	The macro <code>\columnseparator</code> is called between each left/right pair of lines. By default it inserts a vertical rule of width <code>\columnrulewidth</code> . As this is initially defined to be 0pt the rule is invisible. For a visible rule between the columns you could try: <code>\setlength{\columnrulewidth}{0.4pt}</code>
<code>\columnseparator</code>	
	You can also modify <code>\columnseparator</code> if you want more control.
<code>\columnspan</code>	By default, columns are positioned to the right of the page. However, you use <code>\columnspan{L}</code> to align them to the left, or <code>\columnspan{C}</code> to center them.
	When you use <code>\stanza</code> , the visible rule may shift when a verse has a hanging indent. To prevent shifting, use <code>\setstanzaindents</code> outside the <code>Leftside</code> or <code>Rightside</code> environment.
<code>\beforecolumnseparator</code>	By default, the spaces around column separator are the same as the space: <ul style="list-style-type: none"> <li>• On the left of columns, if columns are aligned right.</li> <li>• On the right of columns, if columns are aligned left.</li> <li>• On both the Left and Right columns, if columns are centered.</li> </ul>
<code>\aftercolumnseparator</code>	
	You can redefine <code>\beforecolumnseparator</code> and <code>\aftercolumnseparator</code> length to define spaces before or after the column separator, instead of letting <code>eledpar</code> calculate them automatically. <code>\setlength{\beforecolumnseparator}{length}</code> <code>\setlength{\aftercolumnseparator}{length}</code>
<code>\widthliketwocolumns</code>	If you want to come back to the previous behavior, just set them with a negative value. If you want to mix texts in columns and text without columns, you can horizontally align text in one column to text in two columns with <code>\widthliketwocolumnstrue</code> . To reset this feature, just use <code>\widthliketwocolumnsfalse</code> .
<code>\Xnoteswidthliketwocolumns</code>	In most case, you should use <code>\widthliketwocolumns</code> in combination with <code>\Xnoteswidthliketwocolumns</code> and <code>\notesXwidthliketwocolumns</code> to align the critical/familiar footnotes with the two columns. <code>eledmac</code> handbook for more details.
<code>\notesXwidthliketwocolumns</code>	

## 4 Facing pages

`pages` Numbered text that is to be set on facing pages must be within a `pages` environ-

ment. 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.

`\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.

`\goalfraction` When doing parallel pages `eledpar` has to guess where TeX is going to put pagebreaks 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}
```

## 5 Left and right texts

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

`Leftside` The left text is put within the `Leftside` environment and the right text likewise in the `Rightside` environment. The number of `Leftside` and `Rightside` environments must be the same.

Within these environments you can designate the line numbering scheme(s) to be used. The `eledmac` package originally used counters for specifying the numbering scheme; now both `eledmac`<sup>1</sup> 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

---

<sup>1</sup>when used with `ledpatch` v0.2 or greater.

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

gives you independent control over the left and right numbering schemes. The `\firstsublinenum` and `\sublinenumincrement` macros correspondingly set the numbering scheme for sublines. The starred versions change both left and right numbering schemes.

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

```
\begin{...side}
% \beginnumbering
\pstart first chunk \pend
\pstart  second chunk \pend
...
\pstart  last chunk \pend
% \endnumbering
\end{...side}
```

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

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

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



However, sometime if the left pstarts are much greater than right pstarts, or *vice-versa*, you can decide to shift the pstarts on the left and right side. That means the start of pstarts are not aligned horizontally on the page, the shift is offset at the end of each double pages. To enable this function, load eldpar 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{Leftside}
  \beginnumbering
  ...
\end{Leftside}
\begin{Rightside}
  \beginnumbering
  ...
\end{Rightside}
\Pages
\begin{Leftside}
  \memorydump
  ...
\end{Leftside}
\begin{Rightside}
  \memorydump
  ...
```

`\Rlineflag` The value of `\Rlineflag` is appended to the line numbers of the right texts.

Its default definition is:

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

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

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

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

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

## 7 Verse

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

`astedpar` 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 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}
  ...
\end{pairs}
```

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 &
\end{astanza}
\end{pairs}
```

```

\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  $\text{\LaTeX}$  two time. Example for the French typography

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

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

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

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

## 8 Side notes

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

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

## 9 Parallel ledgroups

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

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

See the following example:

```

\begin{pages}
\begin{Leftside}
\beginnumbering
\pstart
\begin{ledgroup}
    ledgroup content
\end{ledgroup}

```

```

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

```

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

```

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

```

### 9.1 Parallel ledgroups and `setspace` package

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

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

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

## 10 Sectioning commands

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<sup>A</sup>T<sub>E</sub>X marks for header are token from left side. You can change it, using `\eledsectmark{<arg>}`, where `<arg>` could be L (for left side) or R (for right side).

## 11 Implementation overview

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

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

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

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

## 12 Preliminaries

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

```

1 (*code)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{eledpar}[2015/01/12 v1.10.0 eledmac extension for parallel texts]%
4

```

With the option ‘shiftedpstarts’ a long pstart on the left side (or in the right side) don’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 shifted stop at every end of pages. The `\shiftedverses` is kept for backward compatibility.

`\ifshiftedpstarts`

```

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

```

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

```

11 \DeclareOption{widthliketwocolumns}{\widthliketwocolumnstrue}%

```

```

12 \ProcessOptions%

```

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

`\ifl@dpairing` `\ifl@dpairing` is set TRUE if we are processing parallel texts and `\ifl@dpaging` 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`.

```

13 \l@dpairingfalse
14 \l@dpagingfalse
15 \ledRcolfalse

```

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

```

\Lcolwidth
\Rcolwidth
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}}
22 % \end{macrocode}
23 % \end{macro}
24 % \begin{macro}{\led@err@TooManyPstarts}
25 % \begin{macrocode}
26 \newcommand*{\led@err@TooManyPstarts}{%
27 \eledpar@error{Too many \string\pstart\space without printing.
28 \qquad Some text will be lost}{\@ehc}}
```

`\led@err@BadLeftRightPstarts`

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

`\led@err@LeftOnRightPage`

`\led@err@RightOnLeftPage`

```
32 \newcommand*{\led@err@LeftOnRightPage}{%
33 \eledpar@error{The left page has ended on a right page}{\@ehc}}
34 \newcommand*{\led@err@RightOnLeftPage}{%
35 \eledpar@error{The right page has ended on a left page}{\@ehc}}
```

## 13 Sectioning commands

`\section@numR`

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

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

```
36 \newcount\section@numR
37 \section@numR=\z@
```

`\ifpst@rtedL`

`\ifpst@rtedL` is set FALSE at the start of left side numbering, and similarly for

`\ifpst@rtedR`

`\ifpst@rtedR`. `\ifpst@rtedL` is defined in `eledmac`.

```
38 \pst@rtedLfalse
39 \newif\ifpst@rtedR
40 \pst@rtedRfalse
41
```

`\beginnumberingR`

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

```
42 \newcommand*{\beginnumberingR}{%
43 \ifnumberingR
44 \led@err@NumberingStarted
45 \endnumberingR
46 \fi
```



```

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

```

**\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.

```

73 \def\endnumberingR{%
74 \ifnumberingR
75 \global\numberingRfalse
76 \normal@pars
77 \ifl@dpairing
78 \global\pst@rtedRfalse
79 \else
80 \ifx\insertlines@listR\empty\else
81 \global\noteschanged@true
82 \fi
83 \ifx\line@listR\empty\else
84 \global\noteschanged@true
85 \fi
86 \fi
87 \ifnoteschanged@
88 \led@mess@NotesChanged
89 \fi

```

```

90 \else
91   \led@err@NumberingNotStarted
92 \fi
93 \endgroup
94 \if@noeled@sec\else%
95   \immediate\closeout\eled@sectioningR@out%
96 \fi%
97 }
98

```

`\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<sup>A</sup>T<sub>E</sub>X counter in `\numberingR`.

```

99 \newcounter{chapterR}
100 \newcounter{sectionR}
101 \newcounter{subsectionR}
102 \newcounter{subsubsectionR}
103 \newcommand{\initnumbering@sectcountR}{
104   \let\c@chapter\c@chapterR
105   \let\c@section\c@sectionR
106   \let\c@subsection\c@subsectionR
107   \let\c@subsubsection\c@subsubsectionR
108 }

```

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

```

109 \newcommand*{\pausenumberingR}{%
110   \endnumberingR\global\numberingRtrue}
111 \newcommand*{\resumenumberingR}{%
112   \ifnumberingR
113     \global\pst@rtedRtrue
114     \global\advance\section@numR \@ne
115     \led@mess@SectionContinued{\the\section@numR R}%
116     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
117     \l@dend@stuff
118     \begingroup%
119     \initnumbering@sectcountR%
120   \else
121     \led@err@numberingShouldHaveStarted
122     \endnumberingR
123     \beginnumberingR
124   \fi}
125

```

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

```

126 \newcommand*{\memorydumpL}{%
127   \endnumbering
128   \numberingtrue

```

```

129 \global\pst@rtedLtrue
130 \global\advance\section@num \@ne
131 \led@mess@SectionContinued{\the\section@num}%
132 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
133 \l@dend@stuff}
134 \newcommand*{\memorydumpR}{%
135 \endnumberingR
136 \numberingRtrue
137 \global\pst@rtedRtrue
138 \global\advance\section@numR \@ne
139 \led@mess@SectionContinued{\the\section@numR R}%
140 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
141 \l@dend@stuff}
142

```

## 14 Line counting

### 14.1 Choosing the system of lineation

Sometimes you want line numbers that start at 1 at the top of each page; sometimes you want line numbers that start at 1 at each `\pstart`; other times you want line numbers that start at 1 at the start of each section and increase regardless of page breaks. `eledpar` lets you choose different schemes for the left and right texts.

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

`\bypstart@Rtrue`

`\bypstart@Rfalse`

- line-of-page : `bypstart@R = false` and `bypage@R = true`.
- line-of-pstart : `bypstart@R = true` and `bypage@R = false`.

`\ifbypage@R`

`\bypage@Rtrue`

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

```

143 \newif\ifbypage@R
144 \newif\ifbypstart@R
145 \bypage@Rfalse
146 \bypstart@Rfalse

```

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

```

147 \newcommand*{\lineationR}[1]{%
148 \ifnumbering
149 \led@err@LineationInNumbered
150 \else
151 \def\@tempa{#1}\def\@tempb{page}%
152 \ifx\@tempa\@tempb
153 \global\bypage@Rtrue
154 \global\bypstart@Rfalse
155 \else
156 \def\@tempb{pstart}%
157 \ifx\@tempa\@tempb

```

```

158         \global\bypage@Rfalse
159         \global\bystart@Rtrue
160     \else
161         \def@tempb{section}
162         \ifx\@tempa\@tempb
163             \global\bypage@Rfalse
164             \global\bystart@Rfalse
165         \else
166             \led@warn@BadLineation
167         \fi
168     \fi
169 \fi
170 \fi}}

```

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

```

171 \WithSuffix\newcommand\lineation*[1]{%
172   \lineation{#1}%
173   \lineationR{#1}%
174 }%

```

`\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 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.

```

175 \newcount\line@marginR
176 \renewcommand*{\linenummargin}[1]{%
177   \l@getline@margin{#1}%
178   \ifnum\@l@dtmpcntb>\m@ne
179     \ifledRcol
180       \global\line@marginR=\@l@dtmpcntb
181     \else
182       \global\line@margin=\@l@dtmpcntb
183     \fi
184   \fi}}

```

By default put right text numbers at the right.

```

185 \line@marginR=\@ne
186

```

`\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.

```

187 \newcounter{firstlinenumR}

```

```

188 \setcounter{firstlinenumR}{5}
189 \newcounter{linenumincrementR}
190 \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.

```

191 \newcounter{firstsublinenumR}
192 \setcounter{firstsublinenumR}{5}
193 \newcounter{sublinenumincrementR}
194 \setcounter{sublinenumincrementR}{5}
195

```

`\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 `\provide`ing them. The starred  
`\firstsublinenum` versions are specific to eledpar.

```

\sublinenumincrement 196 \providecommand*\firstlinenum{}
\firstlinenum*       197 \providecommand*\linenumincrement{}
\linenumincrement*   198 \providecommand*\firstsublinenum{}
\firstsublinenum*    199 \providecommand*\sublinenumincrement{}
\sublinenumincrement* 200 \renewcommand*\firstlinenum[1]{%
201   \ifledRcol \setcounter{firstlinenumR}{#1}%
202   \else      \setcounter{firstlinenumR}{#1}%
203   \fi}
204 \renewcommand*\linenumincrement[1]{%
205   \ifledRcol \setcounter{linenumincrementR}{#1}%
206   \else      \setcounter{linenumincrementR}{#1}%
207   \fi}
208 \renewcommand*\firstsublinenum[1]{%
209   \ifledRcol \setcounter{firstsublinenumR}{#1}%
210   \else      \setcounter{firstsublinenumR}{#1}%
211   \fi}
212 \renewcommand*\sublinenumincrement[1]{%
213   \ifledRcol \setcounter{sublinenumincrementR}{#1}%
214   \else      \setcounter{sublinenumincrementR}{#1}%
215   \fi}
216 \WithSuffix\newcommand\firstlinenum*[1]{\setcounter{firstlinenumR}{#1}\setcounter{firstlinenum}{#1}}
217 \WithSuffix\newcommand\linenumincrement*[1]{\setcounter{linenumincrementR}{#1}\setcounter{linenumincrement}{#1}}
218 \WithSuffix\newcommand\firstsublinenum*[1]{\setcounter{firstsublinenumR}{#1}\setcounter{firstsublinenum}{#1}}
219 \WithSuffix\newcommand\sublinenumincrement*[1]{\setcounter{sublinenumincrementR}{#1}\setcounter{sublinenumincrement}{#1}}

```

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

```

220 \newcommand*\Rlineflag{R}
221

```

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

```

222 \newcommand*\linenumrepR[1]{\@arabic{#1}}
223 \newcommand*\sublinenumrepR[1]{\@arabic{#1}}
224

```

`\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`.

```

225 \newcommand*{\leftlinenumR}{%
226   \l@dlinenumR
227   \kern\linenumsep}
228 \newcommand*{\rightlinenumR}{%
229   \kern\linenumsep
230   \l@dlinenumR}
231 \newcommand*{\l@dlinenumR}{%
232   \numlabfont\linenumrepR{\line@numR}\Rlineflag%
233   \ifsublines@
234     \ifnum\subline@num>\z@
235       \unskip\fullstop\sublinenumrepR{\subline@numR}%
236     \fi
237   \fi}
238
```

## 14.2 Line-number counters and lists

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

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

```

239 \newcount\line@numR
240 \newcount\subline@numR
241 \newcount\absline@numR
242
```

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

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

```

243 \list@create{\line@listR}
244 \list@create{\insertlines@listR}
245 \list@create{\actionlines@listR}
246 \list@create{\actions@listR}
247 \list@create{\sw@listR}%
248 \list@create{\sw@list@inedtextR}%
249
```

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

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

```

250 \list@create{\linesinpar@listL}
251 \list@create{\linesinpar@listR}
252 \list@create{\maxlinesinpar@list}
253

```

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

```

254 \newcount\page@numR
255

```

### 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.

```

256 \renewcommand*{\read@linelist}[1]{%

```

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

```

257 \ifledRcol
258   \list@clear{\line@listR}%
259   \list@clear{\insertlines@listR}%
260   \list@clear{\actionlines@listR}%
261   \list@clear{\actions@listR}%
262   \list@clear{\linesinpar@listR}%
263   \list@clear{\linesonpage@listR}
264   \list@clear{\sw@listR}%
265   \list@clear{\sw@list@inedtextR}%
266 \else
267   \list@clearing@reg
268   \list@clear{\linesinpar@listL}%
269   \list@clear{\linesonpage@listL}%
270 \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).

```

271 \list@clear{\maxlinesinpar@list}

```

Now get the file and interpret it.

```

272 \get@linelistfile{#1}%
273 \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.

```

274 \ifledRcol
275   \global\page@numR=\m@ne
276   \ifx\actionlines@listR\empty
277     \gdef\next@actionlineR{1000000}%
278   \else
279     \gl@p\actionlines@listR\to\next@actionlineR

```

```

280     \glp\actions@listR\to\next@actionR
281   \fi
282 \else
283   \global\page@num=\m@ne
284   \ifx\actionlines@list\empty
285     \gdef\next@actionline{1000000}%
286   \else
287     \glp\actionlines@list\to\next@actionline
288     \glp\actions@list\to\next@action
289   \fi
290 \fi}
291

```

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.

```

292 \newcommand{\@nl@regR}{%
293   \ifx\l@dchset@num\relax \else
294     \advance\absline@numR \@ne
295     \set@line@action
296     \let\l@dchset@num\relax
297     \advance\absline@numR \m@ne
298     \advance\line@numR \m@ne%    % do we need this?
299   \fi
300   \advance\absline@numR \@ne
301   \ifx\next@page@numR\relax \else
302     \page@action
303     \let\next@page@numR\relax
304   \fi
305   \ifx\sub@change\relax \else
306     \ifnum\sub@change>z@
307       \sublines@true
308     \else
309       \sublines@false
310     \fi
311     \sub@action
312     \let\sub@change\relax
313   \fi

```



```

314 \ifcase\@lockR
315 \or
316   \@lockR \tw@
317 \or\or
318   \@lockR \z@
319 \fi
320 \ifcase\sub@lockR
321 \or
322   \sub@lockR \tw@
323 \or\or
324   \sub@lockR \z@
325 \fi
326 \ifsublines@
327   \ifnum\sub@lockR<\tw@
328     \advance\subline@numR \@ne
329   \fi
330 \else
331   \ifnum\@lockR<\tw@
332     \advance\line@numR \@ne \subline@numR \z@
333   \fi
334 \fi}
335
336 \renewcommand*{\@nl}[2]{%
337   \fix@page{#1}%
338   \ifledRcol
339     \@nl@regR
340   \else
341     \@nl@reg
342   \fi}
343

```

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

```

\fix@page 344 \newcount\last@page@numR
345   \last@page@numR=-10000
346 \renewcommand*{\fix@page}[1]{%
347   \ifledRcol
348     \ifnum #1=\last@page@numR
349     \else
350       \ifbypage@R
351         \line@numR \z@ \subline@numR \z@
352       \fi
353       \page@numR=#1\relax
354       \last@page@numR=#1\relax
355       \def\next@page@numR{#1}%
356     \fi
357   \else
358     \ifnum #1=\last@page@num
359     \else
360       \ifbypage@
361         \line@num \z@ \subline@num \z@

```

```

362     \fi
363     \page@num=#1\relax
364     \last@page@num=#1\relax
365     \def\next@page@num{#1}%
366     \listxadd{\normal@page@break}{\the\absline@num}
367     \fi
368 \fi}
369

```

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

```

370 \renewcommand*{\@adv}[1]{%
371   \ifsublines@
372     \ifledRcol
373       \advance\subline@numR by #1\relax
374       \ifnum\subline@numR<\z@
375         \led@warn@BadAdvancelineSubline
376         \subline@numR \z@
377       \fi
378     \else
379       \advance\subline@num by #1\relax
380       \ifnum\subline@num<\z@
381         \led@warn@BadAdvancelineSubline
382         \subline@num \z@
383       \fi
384     \fi
385   \else
386     \ifledRcol
387       \advance\line@numR by #1\relax
388       \ifnum\line@numR<\z@
389         \led@warn@BadAdvancelineLine
390         \line@numR \z@
391       \fi
392     \else
393       \advance\line@num by #1\relax
394       \ifnum\line@num<\z@
395         \led@warn@BadAdvancelineLine
396         \line@num \z@
397       \fi
398     \fi
399   \fi
400   \set@line@action}
401

```

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

```

402 \renewcommand*{\@set}[1]{%
403   \ifledRcol
404     \ifsublines@

```

```

405     \subline@numR=#1\relax
406   \else
407     \line@numR=#1\relax
408   \fi
409   \set@line@action
410 \else
411   \ifsublines@
412     \subline@num=#1\relax
413   \else
414     \line@num=#1\relax
415   \fi
416   \set@line@action
417 \fi}
418

```

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

```

419 \renewcommand*{\l@d@set}[1]{%
420   \ifledRcol
421     \line@numR=#1\relax
422     \advance\line@numR \@ne
423     \def\l@dchset@num{#1}
424   \else
425     \line@num=#1\relax
426     \advance\line@num \@ne
427     \def\l@dchset@num{#1}
428   \fi}
429 \let\l@dchset@num\relax
430

```

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

```

431 \renewcommand*{\page@action}{%
432   \ifledRcol
433     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
434     \xright@appenditem{\next@page@numR}\to\actions@listR
435   \else
436     \xright@appenditem{\the\absline@num}\to\actionlines@list
437     \xright@appenditem{\next@page@num}\to\actions@list
438   \fi}

```

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

```

439 \renewcommand*{\set@line@action}{%
440   \ifledRcol
441     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
442   \ifsublines@
443     \l@tempcnta=-\subline@numR

```

```

444 \else
445 \l@ldtempcnta=-\line@numR
446 \fi
447 \advance\l@ldtempcnta by -5000\relax
448 \xright@appenditem{\the\l@ldtempcnta}\to\actions@listR
449 \else
450 \xright@appenditem{\the\absline@num}\to\actionlines@list
451 \ifsublines@
452 \l@ldtempcnta=-\subline@num
453 \else
454 \l@ldtempcnta=-\line@num
455 \fi
456 \advance\l@ldtempcnta by -5000\relax
457 \xright@appenditem{\the\l@ldtempcnta}\to\actions@list
458 \fi}
459

```

`\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.

```

460 \renewcommand*{\sub@action}{%
461 \ifledRcol
462 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
463 \ifsublines@
464 \xright@appenditem{-1001}\to\actions@listR
465 \else
466 \xright@appenditem{-1002}\to\actions@listR
467 \fi
468 \else
469 \xright@appenditem{\the\absline@num}\to\actionlines@list
470 \ifsublines@
471 \xright@appenditem{-1001}\to\actions@list
472 \else
473 \xright@appenditem{-1002}\to\actions@list
474 \fi
475 \fi}
476

```

`\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.

```

477 \newcount\@lockR
478 \newcount\sub@lockR
479
480 \newcommand*{\do@lockonR}{%
481 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
482 \ifsublines@
483 \xright@appenditem{-1005}\to\actions@listR
484 \ifnum\sub@lockR=\z@
485 \sub@lockR \@ne

```

```

486     \else
487         \ifnum\sub@lockR=\thr@@
488             \sub@lockR \@ne
489         \fi
490     \fi
491 \else
492     \xright@appenditem{-1003}\to\actions@listR
493     \ifnum\@lockR=\z@
494         \@lockR \@ne
495     \else
496         \ifnum\@lockR=\thr@@
497             \@lockR \@ne
498         \fi
499     \fi
500 \fi}
501
502 \renewcommand*{\do@lockon}{%
503     \ifx\next\lock@off
504         \global\let\lock@off=\skip@lockoff
505     \else
506         \ifledRcol
507             \do@lockonR
508         \else
509             \do@lockonL
510         \fi
511     \fi}

```

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

```

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

```

```

534     \do@lockoffR
535   \else
536     \do@lockoffL
537   \fi}
538 \global\let\lock@off=\do@lockoff
539

```

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

```

540 \providecommand*{\n@num}{%
541 \renewcommand*{\n@num}{%
542   \ifledRcol
543     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
544     \xright@appenditem{-1007}\to\actions@listR
545   \else
546     \n@num@reg
547   \fi}
548

```

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

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

```

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\insert@countR=#1\relax
553     \loop\ifnum\insert@countR>\z@
554       \xright@appenditem{\the\absline@numR}\to\insertlines@listR
555       \global\advance\insert@countR \m@ne
556     \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.

```

557   \begingroup
558     \let\@ref=\dummy@ref

```

```

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

```

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

```

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

```

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

```

575 #2
576 \else

```

And when not in right text

```

577 \@ref@reg{#1}{#2}%
578 \fi}

```

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

```

579 \providecommand*\@pend}[1]{}
580 \renewcommand*\@pend}[1]{%
581 \ifbypstart@\global\line@num=0\fi%
582 \xright@appenditem{#1}\to\linesinpar@listL}
583 \providecommand*\@pendR}[1]{}
584 \renewcommand*\@pendR}[1]{%
585 \ifbypstart@R\global\line@numR=0\fi
586 \xright@appenditem{#1}\to\linesinpar@listR}
587

```

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

```

588 \providecommand*\@lopL}[1]{}
589 \renewcommand*\@lopL}[1]{%
590 \xright@appenditem{#1}\to\linesonpage@listL}
591 \providecommand*\@lopR}[1]{}

```

```

592 \renewcommand*{\@lopR}[1]{%
593   \xright@appenditem{#1}\to\linesonpage@listR}
594

```

## 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`.

```
595 \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 596 \newif\iffirst@linenum@out@R
597   \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.

```

598 \newcommand*{\line@list@stuffR}[1]{%
599   \read@linelist{#1}%
600   \iffirst@linenum@out@R
601     \immediate\closeout\linenum@outR
602     \global\first@linenum@out@Rfalse
603     \immediate\openout\linenum@outR=#1
604   \else
605     \if@minipage%
606       \if@ledgroup%
607         \closeout\linenum@outR
608         \openout\linenum@outR=#1
609       \else
610         \immediate\closeout\linenum@outR
611         \immediate\openout\linenum@outR=#1\relax
612       \fi
613     \else
614       \closeout\linenum@outR%
615       \openout\linenum@outR=#1\relax%
616     \fi
617   \fi}
618

```

`\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.

```

619 \newcommand*{\new@lineL}{%
620   \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.

```

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

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

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

```

623 \renewcommand*{\startsub}{\dimen0\lastskip
624   \ifdim\dimen0>0pt \unskip \fi
625   \ifledRcol \write\linenum@outR{\string\sub@on}%
626   \else      \write\linenum@out{\string\sub@on}%
627   \fi
628   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
629 \def\endsub{\dimen0\lastskip
630   \ifdim\dimen0>0pt \unskip \fi
631   \ifledRcol \write\linenum@outR{\string\sub@off}%
632   \else      \write\linenum@out{\string\sub@off}%
633   \fi
634   \ifdim\dimen0>0pt \hskip\dimen0 \fi}
635
```

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

```

636 \renewcommand*{\advanceline}[1]{%
637   \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
638   \else      \write\linenum@out{\string\@adv[#1]}%
639   \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.

```

640 \renewcommand*{\setline}[1]{%
641   \ifnum#1<\z@
642     \led@warn@BadSetline
643   \else
644     \ifledRcol \write\linenum@outR{\string\@set[#1]}%
645     \else      \write\linenum@out{\string\@set[#1]}%
646     \fi
647   \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.

```

648 \renewcommand*{\setlinenum}[1]{%
649   \ifnum#1<\z@
650     \led@warn@BadSetlinenum
651   \else
652     \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
```

```

653     \else          \write\linenum@out{\string\l@d@set[#1]} \fi
654   \fi}
655

```

`\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.

```

656 \renewcommand*{\startlock}{%
657   \ifledRcol \write\linenum@outR{\string\lock@on}%
658   \else      \write\linenum@out{\string\lock@on}%
659   \fi}
660 \def\endlock{%
661   \ifledRcol \write\linenum@outR{\string\lock@off}%
662   \else      \write\linenum@out{\string\lock@off}%
663   \fi}
664

```

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

```

665 \renewcommand*{\skipnumbering}{%
666   \ifledRcol \write\linenum@outR{\string\n@num}%
667   \advanceline{-1}%
668   \else
669     \skipnumbering@reg
670   \fi}
671

```

## 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` Now we begin `\critext` itself.

We slightly modify the original to make accomodation for when right text is being processed.

```

672 \long\def\critext#1#2/{\leavevmode
673   \@edtext@true%
674   \begingroup
675     \renewcommand{\@tag}{\no@expands #1}%
676     \set@line
677     \ifl@Rcol \global\insert@countR \z@
678     \else      \global\insert@count \z@ \fi
679     \ignorespaces #2\relax
680     \@ifundefined{xpg@main@language}{%if not polyglossia
681       \flag@start}%
682       {\ifRTL\flag@end\else\flag@start\fi% be careful on the direction of writing with polyglossia
683       }%
684   \endgroup
685   \showlemma{#1}%
686   \ifx\end@lemmas\empty \else
687     \gl@p\end@lemmas\to\x@lemma
688     \x@lemma
689     \global\let\x@lemma=\relax
690   \fi
691   \@ifundefined{xpg@main@language}{%if not polyglossia
692     \flag@end}%
693     {\ifRTL\flag@start\else\flag@end\fi% be careful on the direction of writing with polyglossia
694     }
695   \@edtext@false%
696   \@lemmacommand@false%
697 }

```

\edtext And similarly for \edtext.

```

698 \renewcommand{\edtext}[2]{\leavevmode
699   \@edtext@true%
700   \begingroup%
701     \renewcommand{\@tag}{\no@expands #1}%
702     \set@line%
703     \ifl@Rcol \global\insert@countR \z@%
704     \else      \global\insert@count \z@ \fi%
705     \ignorespaces #2\relax%
706     \@ifundefined{xpg@main@language}{%if not polyglossia
707       \flag@start}%
708       {\ifRTL\flag@end\else\flag@start\fi% be careful on the direction of writing with polyglossia
709       }%
710   \endgroup%
711   \showlemma{#1}%
712   \ifx\end@lemmas\empty \else%
713     \gl@p\end@lemmas\to\x@lemma%
714     \x@lemma%
715     \global\let\x@lemma=\relax%
716   \fi%
717   \@ifundefined{xpg@main@language}{%if not polyglossia
718     \flag@end}%
719     {\ifRTL\flag@start\else\flag@end\fi% be careful on the direction of writing with polyglossia

```

```

720     }%
721     \@edtext@false%
722     \@lemmacommand@false%
723 }
724

```

`\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`.

```

725 \renewcommand*{\set@line}{%
726   \ifledRcol
727     \ifx\line@listR\empty
728       \global\noteschanged@true
729       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
730     \else
731       \gl@p\line@listR\to\@tempb
732       \xdef\l@d@nums{\@tempb|\edfont@info}%
733       \global\let\@tempb=\undefined
734     \fi
735   \else
736     \ifx\line@list\empty
737       \global\noteschanged@true
738       \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
739     \else
740       \gl@p\line@list\to\@tempb
741       \xdef\l@d@nums{\@tempb|\edfont@info}%
742       \global\let\@tempb=\undefined
743     \fi
744   \fi}
745

```

## 16 Parallel environments

The initial set up for parallel processing is deceptively simple.

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

`chapterinpages`

```

746 \newenvironment{pairs}{%
747   \l@dpairingtrue
748   \l@dpagingfalse
749   \initnumbering@sectcmd
750   \@atbegin@pairs%
751 }{%
752   \l@dpairingfalse
753 }
754

```

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

```

755 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
756 \def\at@begin@pairs{%
757

```

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.

```

758 \newenvironment{pages}{%
759 \let\oldchapter\chapter
760 \let\chapter\chapterinpages
761 \l@dpairingtrue
762 \l@dpagingtrue
763 \initnumbering@sectcmd
764 \setlength{\Lcolwidth}{\textwidth}%
765 \setlength{\Rcolwidth}{\textwidth}%
766 }{%
767 \l@dpairingfalse
768 \l@dpagingfalse
769 \let\chapter\oldchapter
770 }
771 \newcommand{\chapterinpages}{\thispagestyle{plain}}%
772 \global\@topnum\z@
773 \@afterindentfalse
774 \secdef\@chapter\@schapter}
775

```

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

```

776 \newif\ifinstanzaL
777 \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.

```

778 \newenvironment{Leftside}{%
779 \ledRcolfalse
780 \setcounter{pstartL}{1}
781 \let\pstart\pstartL
782 \let\thepstart\thepstartL
783 \let\pend\pendL
784 \let\memorydump\memorydumpL
785 \Leftsidehook
786 \let\old@startstanza\@startstanza
787 \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
788 }{
789 \Leftsidehookend}

```

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

`\Leftsidehookend`

`\Rightsidehook` 790 `\newcommand*{\Leftsidehook}{}`

`\Rightsidehookend` 791 `\newcommand*{\Leftsidehookend}{}`

792 `\newcommand*{\Rightsidehook}{}`

793 `\newcommand*{\Rightsidehookend}{}`

794

`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.

```
795 \newenvironment{Rightside}{%
796   \ledRcoltrue
797   \let\beginnumbering\beginnumberingR
798   \let\endnumbering\endnumberingR
799   \let\pausenumbering\pausenumberingR
800   \let\resumenumbering\resumenumberingR
801   \let\memorydump\memorydumpR
802   \let\thepstart\thepstartR
803   \let\pstart\pstartR
804   \let\pend\pendR
805   \let\ledpb\ledpbR
806   \let\lednopb\lednopbR
807   \let\lineation\lineationR
808   \Rightsidehook
809   \let\old@startstanza\@startstanza
810   \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
811 }{%
812   \ledRcolfalse
813   \Rightsidehookend
814 }
815
```

## 17 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.

### 17.1 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.

```
816 \newcount\num@linesR
817 \newbox\one@lineR
818 \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.

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. The `old` counters are used to have the good reset of the `pstart` counters at the beginning of the `\Pages` command.

```
819
820 \newcounter{pstartL}
821 \newcounter{pstartLold}
822 \renewcommand{\thepstartL}{\bfseries\@arabic{c@pstartL}. }
823 \newcounter{pstartR}
824 \newcounter{pstartRold}
825 \renewcommand{\thepstartR}{\bfseries\@arabic{c@pstartR}. }
826
827 \newcommandx*{\pstartL}[1][1]{%
828   \if@nobreak%
829     \let\@oldnobreak\@nobreaktrue%
830   \else%
831     \let\@oldnobreak\@nobreakfalse%
832   \fi%
833   \@nobreaktrue%
834   \ifnumbering \else%
835     \led@err@PstartNotNumbered%
836     \beginnumbering%
837   \fi%
838   \ifnumberedpar@%
839     \led@err@PstartInPstart%
840   \pend%
841 \fi%
```

If this is the first `\pstart` in a numbered section, clear any inserts and set `\ifpstart@rtedL` to `FALSE`. Save the `pstartL` counter.

```

842 \ifpst@rtedL\else%
843   \setcounter{pstartLold}{\value{pstartL}}%
844   \list@clear{\inserts@list}%
845   \global\let\next@insert=\empty%
846   \global\pst@rtedLtrue%
847 \fi%
848 \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.

```

849 \global\advance\l@dnumpststartsL \@ne%
850 \ifnum\l@dnumpststartsL>\l@dc@maxchunks%
851   \led@err@TooManyPstarts%
852   \global\l@dnumpststartsL=\l@dc@maxchunks%
853 \fi%
854 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpststartsL}=\vbox\bgroup%
855 \ifautopar\else%
856   \ifnumberpstart%
857     \ifsidepstartnum%
858       \else%
859         \thepstartL%
860       \fi%
861     \fi%
862   \fi%
863 \hsize=\Lcolwidth%
864 \numberedpar@true%
865 \iflabelpstart\protected@edef\@currentlabel%
866   {\p@pstartL\thepstartL}\fi%

```

Dump the optional arguments

```

867 \ifstrempy{#1}%
868   {\csgdef{before@pstartL@the\l@dnumpststartsL}{\at@every@pstart}}%
869   {\csgdef{before@pstartL@the\l@dnumpststartsL}{\noindent#1}}%
870 }

871 \newcommandx*{\pstartR}[1][1]{%
872   \if@nobreak%
873     \let\@oldnobreak\@nobreaktrue%
874   \else%
875     \let\@oldnobreak\@nobreakfalse%
876   \fi%
877   \@nobreaktrue%
878   \ifnumberingR \else%
879     \led@err@PstartNotNumbered%
880     \beginnumberingR%
881   \fi%
882   \ifnumberedpar@%
883     \led@err@PstartInPstart%
884     \pendR%
885   \fi%
886 \ifpst@rtedR\else%

```



```

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

```

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

```

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

```

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

```

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

```

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

```

924 \endgroup%
925 \ignorespaces%
926 \@oldnobreak%

```

```

927 \ifnumberpstart%
928   \addtocounter{pstartL}{1}%
929 \fi
930 \parledgroup@beforenotes@save{L}%
    Dump content of the optional argument.
931 \ifstrempy{#1}%
932   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
933   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
934 }

```

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

```

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

```

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

```

957 \newif\ifprint@last@after@pendL%
958 \newif\ifprint@last@after@pendR%

```

## 17.2 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  
`\l@drightbox` of right text.

```

959 \newbox\l@dleftbox
960 \newbox\l@drightbox
961

```

```

\countLline We need to know the number of lines processed.
\countRline 962 \newcount\countLline
              963 \countLline \z@
              964 \newcount\countRline
              965 \countRline \z@
              966

\@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 967 \newcount\@donereallinesL
                  968 \newcount\@donetotallinesL
                  969 \newcount\@donereallinesR
                  970 \newcount\@donetotallinesR
                  971

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

          972 \newcommand*{\do@lineL}{%
          973 \advance\countLline \@ne
          974 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}%
          975 {\vbadness=10000
          976 \splittopskip=\z@
          977 \do@lineLhook
          978 \l@demtyd@ta
          979 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscL}
          980 to\baselineskip}%
          981 \IfStrEq{\splitfirstmarks\parledgroup@}{\begin}{\parledgroup@notes@startL}{\}
          982 \unvbox\one@line \global\setbox\one@line=\lastbox
          983 \getline@numL
          984 \ifnum\@lock>\@ne%
          985 \inserthangingsymboltrue%
          986 \else%
          987 \inserthangingsymbolfalse%
          988 \fi
          989 \setbox\l@dleftbox
          990 \hb@xt@ \Lcolwidth{%
          991 \affixline@num
          992 \xifinlist{\the\l@dpscL}{\eled@sections@@}%
          993 {\add@inserts\affixside@note}%
          994 {\print@lineL}}%
          995 \add@penaltiesL
          996 \global\advance\@donereallinesL\@ne
          997 \global\advance\@donetotallinesL\@ne
          998 \else
          999 \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
          1000 \global\advance\@donetotallinesL\@ne
          1001 \fi}

```

1002  
1003

`\print@eledsectionL` `\print@lineL` is for lines without a sectioning command.

```
1004 \def\print@lineL{%
1005     \affixpstart@numL%
1006     \l@dld@ta %space kept for backward compatibility
1007     \add@inserts\affixside@note%
1008     \l@dlsn@te %space kept for backward compatibility
1009     {\ledllfill\hb@xt@ \wd\one@line{\do@insidelinelhook\inserthangingsymbolL\new@lineL\l@
1010         \l@drsn@te}}
1011
```

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

```
1012 \def\print@eledsectionL{%
1013     \addtocounter{pstartL}{-1}%
1014     \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
1015     \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}%
1016     \numdef{\temp@}{\l@dpscL-1}%
1017     \xifinlist{\temp@}{\eled@sections@}{\@nobreaktrue}{\@nobreakfalse}%
1018     \@eled@sectioningtrue%
1019     \csuse{eled@sectioning@the\l@dpscL}%
1020     \@eled@sectioningfalse%
1021     \global\csundef{eled@sectioning@the\l@dpscL}%
1022     \if@RTL%
1023         \hspace{-3\paperwidth}%
1024     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1025     \else%
1026         \hspace{3\paperwidth}%
1027     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1028     \fi%
1029     \vskip\eledsection@correcting@skip%
1030 }
1031
```

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

```
\doinsidelinelhook 1032 \newcommand*\dolineLhook[1]{\gdef\do@lineLhook{#1}}%
\doinsidelinerhook 1033 \newcommand*\dolineRhook[1]{\gdef\do@lineRhook{#1}}%
1034 \newcommand*\doinsidelinelhook[1]{\gdef\do@insidelinelhook{#1}}%
1035 \newcommand*\doinsidelinerhook[1]{\gdef\do@insidelinerhook{#1}}%
1036
```

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

```
\do@lineRhook 1037 \newcommand*\do@lineLhook{}
\do@insidelinelhook 1038 \newcommand*\do@lineRhook{}
\do@insidelinerhook 1039 \newcommand*\do@insidelinelhook{}
1040 \newcommand*\do@insidelinerhook{}
1041
```

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

```

1042 \newcommand*{\do@lineR}{%
1043   \ledRcol@true%
1044   \advance\countRline \@ne
1045   \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}%
1046   {\vbadness=10000
1047    \splittopskip=\z@
1048    \do@lineRhook
1049    \l@demtyd@ta
1050    \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscR}
1051                      to\baselineskip}%
1052   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startR}{ }
1053   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox
1054   \getline@numR
1055   \ifnum\@lockR>\@ne%
1056     \inserthangingsymbolRtrue
1057   \else%
1058     \inserthangingsymbolRfalse%
1059   \fi%
1060   \setbox\l@dtrightbox
1061   \hb@xt@ \Rcolwidth{%
1062     \affixline@numR%
1063     \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1064     {\add@insertsR\affixside@noteR}%
1065     {\print@lineR}%
1066   }%
1067   \add@penaltiesR
1068   \global\advance\@donereallinesR\@ne
1069   \global\advance\@donetotallinesR\@ne
1070 \else
1071   \setbox\l@dtrightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}
1072   \global\advance\@donetotallinesR\@ne
1073 \fi
1074 \ledRcol@false%
1075 }
1076
1077

```

`\print@lineR`

`\print@eledsectionR`

### 17.3 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.

```

1078 \newcommand*{\getline@numR}{%
1079   \global\advance\absline@numR \@ne
1080   \do@actionsR
1081   \do@ballastR

```

```

1082 \ifledgroupnotesR@\else\ifnumberline
1083   \ifsublines@
1084     \ifnum\sub@lockR<\tw@
1085       \global\advance\subline@numR \@ne
1086     \fi
1087   \else
1088     \ifnum\@lockR<\tw@
1089       \global\advance\line@numR \@ne
1090       \global\subline@numR \z@
1091     \fi
1092   \fi
1093 \fi
1094 \fi
1095 }
1096 \newcommand*{\getline@numL}{%
1097   \global\advance\absline@num \@ne
1098   \do@actions
1099   \do@ballast
1100 \ifledgroupnotesL@\else\ifnumberline
1101   \ifsublines@
1102     \ifnum\sub@lock<\tw@
1103       \global\advance\subline@num \@ne
1104     \fi
1105   \else
1106     \ifnum\@lock<\tw@
1107       \global\advance\line@num \@ne
1108       \global\subline@num \z@
1109     \fi
1110   \fi
1111 \fi
1112 \fi
1113 }
1114
1115

```

`\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.

```

1116 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1117   \begingroup
1118     \advance\absline@numR \@ne
1119     \ifnum\next@actionlineR=\absline@numR
1120       \ifnum\next@actionR>-1001
1121         \global\advance\ballast@count by -\c@ballast
1122       \fi
1123     \fi
1124   \endgroup}

```

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

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

```

1125 \newcommand*{\do@actions@fixedcodeR}{%
1126   \ifcase\@l@dtmpcnta%
1127   \or%                % 1001
1128     \global\sublines@true
1129   \or%                % 1002
1130     \global\sublines@false
1131   \or%                % 1003
1132     \global\@lockR=\@ne
1133   \or%                % 1004
1134     \ifnum\@lockR=\tw@
1135       \global\@lockR=\thr@@
1136     \else
1137       \global\@lockR=\z@
1138     \fi
1139   \or%                % 1005
1140     \global\sub@lockR=\@ne
1141   \or%                % 1006
1142     \ifnum\sub@lockR=\tw@
1143       \global\sub@lockR=\thr@@
1144     \else
1145       \global\sub@lockR=\z@
1146     \fi
1147   \or%                % 1007
1148     \l@dskipnumbertrue
1149   \else
1150     \led@warn@BadAction
1151   \fi}
1152
1153
1154 \newcommand*{\do@actionsR}{%
1155   \global\let\do@actions@nextR=\relax
1156   \@l@dtmpcntb=\absline@numR
1157   \ifnum\@l@dtmpcntb<\next@actionlineR\else
1158     \ifnum\next@actionR>-1001\relax
1159       \global\page@numR=\next@actionR
1160       \ifbypage@R
1161         \global\line@numR \z@ \global\subline@numR \z@
1162       \fi
1163     \else
1164       \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1165         \@l@dtmpcnta=-\next@actionR
1166         \advance\@l@dtmpcnta by -5001\relax
1167         \ifsublines@
1168           \global\subline@numR=\@l@dtmpcnta
1169         \else
1170           \global\line@numR=\@l@dtmpcnta
1171         \fi

```

```

1172     \else
1173         \@l@dttempcnta=-\next@actionR
1174         \advance\@l@dttempcnta by -1000\relax
1175         \do@actions@fixedcodeR
1176     \fi
1177 \fi
1178 \ifx\actionlines@listR\empty
1179     \gdef\next@actionlineR{1000000}%
1180 \else
1181     \glp\actionlines@listR\to\next@actionlineR
1182     \glp\actions@listR\to\next@actionR
1183     \global\let\do@actions@nextR=\do@actionsR
1184 \fi
1185 \fi
1186 \do@actions@nextR}
1187

```

## 17.4 Line number printing

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

```

\ch@cksub@l@ckR 1188
\ch@ck@l@ckR 1189 \providecommand*{\l@dcalcnun}[3]{%
\fx@l@cksR 1190 \ifnum #1 > #2\relax
\affixline@numR 1191 \l@dttempcnta = #1\relax
1192 \advance\@l@dttempcnta by -#2\relax
1193 \divide\@l@dttempcnta by #3\relax
1194 \multiply\@l@dttempcnta by #3\relax
1195 \advance\@l@dttempcnta by #2\relax
1196 \else
1197 \l@dttempcnta=#2\relax
1198 \fi}
1199
1200 \newcommand*{\ch@cksub@l@ckR}{%
1201 \ifcase\sub@lockR
1202 \or
1203 \ifnum\sublock@disp=\@ne
1204 \l@dttempcntb \z@ \l@dttempcnta \@ne
1205 \fi
1206 \or
1207 \ifnum\sublock@disp=\tw@
1208 \else
1209 \l@dttempcntb \z@ \l@dttempcnta \@ne
1210 \fi
1211 \or
1212 \ifnum\sublock@disp=\z@
1213 \l@dttempcntb \z@ \l@dttempcnta \@ne
1214 \fi
1215 \fi}
1216

```



```

1217 \newcommand*{\ch@ck@l@ckR}{%
1218   \ifcase\@lockR
1219   \or
1220     \ifnum\lock@disp=\@ne
1221       \@l@tempcntb \z@ \l@tempcnta \@ne
1222     \fi
1223   \or
1224     \ifnum\lock@disp=\tw@
1225     \else
1226       \@l@tempcntb \z@ \l@tempcnta \@ne
1227     \fi
1228   \or
1229     \ifnum\lock@disp=\z@
1230       \@l@tempcntb \z@ \l@tempcnta \@ne
1231     \fi
1232   \fi}
1233
1234 \newcommand*{\f@x@l@ckR}{%
1235   \ifcase\@lockR
1236   \or
1237     \global\@lockR \tw@
1238   \or \or
1239     \global\@lockR \z@
1240   \fi
1241   \ifcase\sub@lockR
1242   \or
1243     \global\sub@lockR \tw@
1244   \or \or
1245     \global\sub@lockR \z@
1246   \fi}
1247
1248
1249 \newcommand*{\affixline@numR}{%
1250 \ifledgroupnotesR\else\ifnumberline
1251 \ifl@dskipnumber
1252   \global\l@dskipnumberfalse
1253 \else
1254   \ifsublines@
1255     \@l@tempcntb=\subline@numR
1256     \l@dcalcnm{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR}%
1257     \ch@cksub@lockR
1258   \else
1259     \@l@tempcntb=\line@numR
1260     \ifx\linenumberlist\empty
1261       \l@dcalcnm{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1262     \else
1263       \@l@tempcnta=\line@numR
1264       \edef\rem@inder{\linenumberlist,\number\line@numR,%
1265         \edef\sc@n@list{\def\noexpand\sc@n@list
1266           ###1,\number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%

```

```

1267 \sc@n@list\expandafter\sc@n@list\rem@inder|%
1268 \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1269 \fi
1270 \ch@ck@l@ckR
1271 \fi
1272 \ifnum\@l@tempcnta=\@l@tempcntb
1273 \if@twocolumn
1274 \if@firstcolumn
1275 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1276 \else
1277 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1278 \fi
1279 \else
1280 \l@tempcntb=\line@marginR
1281 \ifnum\@l@tempcntb>\@ne
1282 \advance\@l@tempcntb by\page@numR
1283 \fi
1284 \ifodd\@l@tempcntb
1285 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1286 \else
1287 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1288 \fi
1289 \fi
1290 \fi
1291 \f@x@l@cksR
1292 \fi
1293 \fi
1294 \fi}

```

## 17.5 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 1295
\leftpstartnumR 1296 \newcommand*{\affixpstart@numL}{%
\rightpstartnumR 1297 \ifsidepstartnum
\leftpstartnumL 1298 \if@twocolumn
\rightpstartnumL 1299 \if@firstcolumn
\ifpstartnumR 1300 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1301 \else
1302 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1303 \fi
1304 \else

```

```

1305 \l@tempcntb=\line@margin
1306 \ifnum\l@tempcntb>\@ne
1307 \advance\l@tempcntb \page@num
1308 \fi
1309 \ifodd\l@tempcntb
1310 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1311 \else
1312 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1313 \fi
1314 \fi
1315 \fi
1316 }
1317 \newcommand*{\affixpstart@numR}{%
1318 \ifsidepstartnum
1319 \if@twocolumn
1320 \if@firstcolumn
1321 \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1322 \else
1323 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1324 \fi
1325 \else
1326 \l@tempcntb=\line@marginR
1327 \ifnum\l@tempcntb>\@ne
1328 \advance\l@tempcntb \page@numR
1329 \fi
1330 \ifodd\l@tempcntb
1331 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1332 \else
1333 \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1334 \fi
1335 \fi
1336 \fi
1337 }
1338
1339 \newcommand*{\leftpstartnumL}{
1340 \ifpstartnum
1341 \thepstartL
1342 \kern\linenumsep\global\pstartnumfalse\fi
1343 }
1344 \newcommand*{\rightpstartnumL}{
1345 \ifpstartnum\kern\linenumsep
1346 \thepstartL
1347 \global\pstartnumfalse\fi
1348 }
1349 \newif\ifpstartnumR
1350 \pstartnumRtrue
1351 \newcommand*{\leftpstartnumR}{
1352 \ifpstartnumR
1353 \thepstartR
1354 \kern\linenumsep\global\pstartnumRfalse\fi

```

```

1355 }
1356 \newcommand*{\rightpstartnumR}{
1357 \ifpstartnumR\kern\linenumsep
1358 \thepstartR
1359 \global\pstartnumRfalse\fi
1360 }

```

## 17.6 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.

```

1361 \list@create{\inserts@listR}

```

`\add@insertsR` The right text version.

```

\add@inserts@nextR 1362 \newcommand*{\add@insertsR}{%
1363   \global\let\add@inserts@nextR=\relax
1364   \ifx\inserts@listR\empty \else
1365     \ifx\next@insertR\empty
1366       \ifx\insertlines@listR\empty
1367         \global\noteschanged@true
1368         \gdef\next@insertR{100000}%
1369       \else
1370         \gl@p\insertlines@listR\to\next@insertR
1371       \fi
1372     \fi
1373     \ifnum\next@insertR=\absline@numR
1374       \gl@p\inserts@listR\to\@insertR
1375       \@insertR
1376       \global\let\@insertR=\undefined
1377       \global\let\next@insertR=\empty
1378       \global\let\add@inserts@nextR=\add@insertsR
1379     \fi
1380   \fi
1381   \add@inserts@nextR}
1382

```

## 17.7 Penalties

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

In the code below, which is a virtual copy of the original `\add@penalties`, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we're working on at the moment. The count `\@l@dttempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`,

which has been worked out in `\do@ballast`. Finally, the penalty is checked to see that it doesn't go below  $-10000$ .

```
\newcommand*{\add@penaltiesR}{\@l@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@tempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@tempcnta=\z@
\relax
\else
\ifnum\@l@tempcnta>-10000
\penalty\@l@tempcnta
\else
\penalty -10000
\fi
\fi}
```

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

```
1383 \newcommand*{\add@penaltiesL}{\}
1384 \newcommand*{\add@penaltiesR}{\}
1385
```

## 17.8 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.

```
1386 \newcommand*{\flush@notesR}{%
1387 \xloop
1388 \ifx\inserts@listR\empty \else
1389 \glp\inserts@listR\to\@insertR
1390 \@insertR
1391 \global\let\@insertR=\undefined
1392 \repeat}
1393
```

## 18 Footnotes

### 18.1 Normal footnote formatting

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

```

\printlinesR This is the right text version of \printlines and takes account of \Rlineflag.
\ledsavedprintlines Just in case, \ledsavedprintlines is a copy of the original \printlines.
                Just a reminder of the arguments:
\printlinesR   #1      |  #2  |   #3   |   #4   |   #5   |   #6   |   #7
\printlinesR start-page | line | subline | end-page | line | subline | font
1394 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1395   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1396   \ifl@d@pnum #1\fullstop\fi
1397   \ifledplinum \linenumr@p{#2}\Rlineflag\else \symplinenum\fi
1398   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1399   \ifl@d@dash \endashchar\fi
1400   \ifl@d@pnum #4\fullstop\fi
1401   \ifl@d@elin \linenumr@p{#5}\Rlineflag\fi
1402   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1403 \endgroup}
1404
1405 \let\ledsavedprintlines\printlines
1406

```

## 19 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.

```

1407 \list@create{\labelref@listR}
1408

```

`\edlabel` The `\edlabel` command first writes a `\@lab` macro to the `\linenum@out` file. It then checks to see that the `\labelref@list` actually has something in it (if not, it creates a dummy entry), and pops the next value for the current label, storing it in `\label@refs`. Finally it defines the label to be `\empty` so that any future check will turn up the fact that it has been used.

```

1409 \renewcommand*{\edlabel}[1]{%
1410   \ifl@d@pairing\ifautopar%
1411     \strut%
1412   \fi\fi%
1413   \@bsphack%
1414   \ifledRcol

```

```

1415 \write\linenum@outR{\string\@lab}%
1416 \ifx\labelref@listR\empty
1417 \xdef\label@refs{\zz@@@}%
1418 \else
1419 \gl@p\labelref@listR\to\label@refs
1420 \fi
1421 \ifvmode
1422 \advancelabel@refs
1423 \fi
1424 \protected@write\@auxout{%
1425 {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR|{#1}}%
1426 \else
1427 \write\linenum@out{\string\@lab}%
1428 \ifx\labelref@list\empty
1429 \xdef\label@refs{\zz@@@}%
1430 \else
1431 \gl@p\labelref@list\to\label@refs
1432 \fi
1433 \ifvmode
1434 \advancelabel@refs
1435 \fi
1436 \protected@write\@auxout{%
1437 {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstart|{#1}}%
1438 \fi
1439 \@esphack}
1440
1441

```

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

```

1442 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1443 \expandafter\ifx\csname the@label#5\endcsname \relax\else
1444 \led@warn@DuplicateLabel{#4}%
1445 \fi
1446 \expandafter\gdef\csname the@label#5\endcsname{#1|#2\Rlineflag|#3|#4}%
1447 \ignorespaces}
1448 \AtBeginDocument{%
1449 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1450 }
1451

```

`\@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.

```

1452 \renewcommand*{\@lab}{%
1453 \ifledRcol
1454 \xright@appenditem{\linenumr@p{\line@numR}|%
1455 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
1456 \to\labelref@listR

```

```

1457 \else
1458   \xright@appenditem{\linenumr@p{\line@num}}|{%
1459     \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
1460   \to\labelref@list
1461 \fi}
1462

```

## 20 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* 1463 \WithSuffix\newcommand\sidenotemargin*[1]{%
1464   \l@dgetsidenote@margin{#1}
1465   \global\sidenote@marginR=\@l@tempcntb
1466   \global\sidenote@margin=\@l@tempcntb
1467 }
1468 \newcount\sidenote@marginR
1469 \global\sidenote@margin=\@ne
1470

```

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

```

1471 \newcommand*\affixside@noteR{%
1472   \def\sidenotecontent@{}%
1473   \numgdef{\itemcount@}{0}%
1474   \def\do##1{%
1475     \ifnumequal{\itemcount@}{0}%
1476     {%
1477       \appto\sidenotecontent@{##1}}% Not print not separator before the 1st note
1478     {\appto\sidenotecontent@{\sidenotesep ##1}%
1479     }%
1480     \numgdef{\itemcount@}{\itemcount@+1}%
1481   }%
1482   \dolistloop{\l@dcnotes@text}%
1483   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1484   \gdef\@templ@d{%
1485     \gdef\@templ@n{\l@dcnotes@text\l@dcnotes@text@l\l@dcnotes@text@r}%
1486     \ifx\@templ@d\@templ@n \else%
1487       \if@twocolumn%
1488         \if@firstcolumn%
1489           \setl@dlp@rbox{##1}{\sidenotecontent@}%
1490         \else%
1491           \setl@drp@rbox{\sidenotecontent@}%
1492         \fi%
1493       \else%
1494         \l@tempcntb=\sidenote@marginR%
1495         \ifnum\l@tempcntb>\@ne%

```



```

1496     \advance\@l@tempcntb by\page@numR%
1497     \fi%
1498     \ifodd\@l@tempcntb%
1499         \setl@drp@rbox{\sidenotecontent@}%
1500         \gdef\sidenotecontent@{}%
1501         \numdef{\itemcount@}{0}%
1502         \dolistloop{\l@dcstetext@l}%
1503         \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1504         \setl@dlp@rbox{\sidenotecontent@}%
1505     \else%
1506         \setl@dlp@rbox{\sidenotecontent@}%
1507         \gdef\sidenotecontent@{}%
1508         \numdef{\itemcount@}{0}%
1509         \dolistloop{\l@dcstetext@r}%
1510         \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1511         \setl@drp@rbox{\sidenotecontent@}%
1512     \fi%
1513 \fi%
1514 \fi%
1515 }
1516

```

## 21 Familiar footnotes

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

```

1517 \renewcommand{\l@dbfnote}[1]{%
1518     \ifnumberedpar@
1519     \gdef\@tag{#1}%
1520     \ifledRcol%
1521         \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@thefnmark}}%
1522         \to\inserts@listR
1523     \global\advance\insert@countR \@ne%
1524 \else%
1525     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@thefnmark}}%
1526     \to\inserts@list
1527     \global\advance\insert@count \@ne%
1528 \fi
1529 \fi\ignorespaces}
1530

```

`\normalbfnoteX`

```

1531 \renewcommand{\normalbfnoteX}[2]{%
1532     \ifnumberedpar@
1533     \ifledRcol%
1534     \ifluatex
1535         \footnotelang@lua[R]%
1536     \fi

```

```

1537 \ifundefined{xpg@main@language}%if polyglossia
1538 {}%
1539 {\footnotelang@poly[R]}%
1540 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1541 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}}{\expandonce\thisfootnote}}%
1542 \to\inserts@listR
1543 \global\advance\insert@countR \@ne%
1544 \else%
1545 \ifluatex
1546 \footnotelang@lua%
1547 \fi
1548 \ifundefined{xpg@main@language}%if polyglossia
1549 {}%
1550 {\footnotelang@poly}%
1551 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
1552 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}}{\expandonce\thisfootnote}}%
1553 \to\inserts@list
1554 \global\advance\insert@count \@ne%
1555 \fi
1556 \fi\ignorespaces}
1557

```

## 22 Verse

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

```

\inserthangingsymbolL
\inserthangingsymbolR 1558 \newif\ifinserthangingsymbolR
1559 \newcommand{\inserthangingsymbolL}{%
1560 \ifinserthangingsymbol%
1561 \ifinstanzaL%
1562 \hangingsymbol%
1563 \fi%
1564 \fi}
1565 \newcommand{\inserthangingsymbolR}{%
1566 \ifinserthangingsymbolR%
1567 \ifinstanzaR%
1568 \hangingsymbol%
1569 \fi%
1570 \fi}

```

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

```

\correcthangingL
\correcthangingR 1571 \newcommand{\correcthangingL}{%

```

```

1572 \ifl@dpaging\else%
1573   \ifinstanzaL%
1574     \ifinserthangingsymbol%
1575       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1576         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1577     \fi%
1578   \fi%
1579 \fi}
1580
1581 \newcommand{\correcthangingR}{%
1582 \ifl@dpaging\else%
1583   \ifinstanzaR%
1584     \ifinserthangingsymbolR%
1585       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1586         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1587     \fi%
1588   \fi%
1589 \fi}

```

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

```

1590 \chardef\next=\catcode'\&
1591 \catcode'\&=\active
1592

```

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

```

1593 \newenvironment{astanza}{%
1594   \startstanzahook
1595   \catcode'\&=\active
1596   \global\stanza@count\@ne\stanza@modulo\@ne
1597   \ifnum\usernamecount{sza@00}=\z@
1598     \let\stanza@hang\relax
1599     \let\endlock\relax
1600   \else
1601   %%% \interlinepenalty\@M % this screws things up, but I don't know why
1602   \rightskip\z@ plus 1fil\relax
1603   \fi
1604   \ifnum\usernamecount{szp@00}=\z@
1605     \let\sza@penalty\relax
1606   \fi
1607   \def&{%
1608     \endlock\mbox{}}%
1609     \sza@penalty
1610     \global\advance\stanza@count\@ne
1611     \@astanza@line}%
1612 \def\&{%
1613   \endlock\mbox{}}

```

```

1614 \pend
1615 \endstanzaextra}%
1616 \pstart
1617 \@astanza@line
1618 }{ }
1619

```

`\@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.

```

1620 \newcommand*{\@astanza@line}{%
1621 \ifnum\value{stanzaindentrepetition}=0
1622 \parindent=\csname sza@number\stanza@count
1623 @\endcsname\stanzaindentbase
1624 \else
1625 \parindent=\csname sza@number\stanza@modulo
1626 @\endcsname\stanzaindentbase
1627 \managestanza@modulo
1628 \fi
1629 \par
1630 \stanza@hang%\mbox{}%
1631 \ignorespaces}
1632

```

Lastly reset the modified category codes.

```

1633 \catcode'\&=\next
1634

```

## 23 Naming macros

The L<sup>A</sup>T<sub>E</sub>X kernel provides `\@namedef` and `\@namuse` for defining and using macros that may have non-letters in their names. We need something similar here as we are going to need and use some numbered boxes and counters.

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

```

\unhnamebox 1635 \providecommand*{\newnamebox}[1]{%
\unvnamebox 1636 \expandafter\newbox\csname #1\endcsname}
\namebox 1637 \providecommand*{\setnamebox}[1]{%
1638 \expandafter\setbox\csname #1\endcsname}
1639 \providecommand*{\unhnamebox}[1]{%
1640 \expandafter\unhbox\csname #1\endcsname}
1641 \providecommand*{\unvnamebox}[1]{%
1642 \expandafter\unvbox\csname #1\endcsname}
1643 \providecommand*{\namebox}[1]{%
1644 \csname #1\endcsname}
1645

```

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

`\usenamecount`

```

1646 \providecommand*\newnamecount}[1]{%
1647   \expandafter\newcount\csname #1\endcsname}
1648 \providecommand*\usernamecount}[1]{%
1649   \csname #1\endcsname}
1650

```

## 24 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.

```

1651 \newcount\l@dc@maxchunks
1652 \newcommand*\maxchunks}[1]{\l@dc@maxchunks=#1}
1653   \maxchunks{5120}
1654

```

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

```

\l@dnumpstartsR 1655 \newcount\l@dnumpstartsR
1656

```

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

```

\l@pscl 1657 \newcount\l@dpscl
1658 \newcount\l@dpsclR
1659

```

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

```

1660 \newcommand*\l@dsetuprawboxes{%
1661   \@l@tempcntb=\l@dc@maxchunks
1662   \loop\ifnum\@l@tempcntb>\z@
1663     \newnamebox{\l@dLcolrawbox\the\@l@tempcntb}
1664     \newnamebox{\l@dRcolrawbox\the\@l@tempcntb}
1665     \advance\@l@tempcntb \m@ne
1666   \repeat}
1667

```

`\l@dsetupmaxlinecounts` 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@dsetupmaxlinecounts` creates `\maxchunks` new counts called `\l@dmaxlinesinpar1`, etc., and `\l@dzeromaxlinecounts` zeroes all of them.

```

1668 \newcommand*\l@dsetupmaxlinecounts{%
1669   \@l@tempcntb=\l@dc@maxchunks
1670   \loop\ifnum\@l@tempcntb>\z@

```

```

1671 \newnamecount{l@dmaxlinesinpar\the\@l@tempcntb}
1672 \advance\@l@tempcntb \m@ne
1673 \repeat}
1674 \newcommand*{\l@dzeromaxlinecounts}{%
1675 \begingroup
1676 \l@tempcntb=\l@dc@maxchunks
1677 \loop\ifnum\@l@tempcntb>\z@
1678 \global\usenamecount{l@dmaxlinesinpar\the\@l@tempcntb}=\z@
1679 \advance\@l@tempcntb \m@ne
1680 \repeat
1681 \endgroup}
1682

```

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

```

1683 \AtBeginDocument{%
1684 \l@dsetupprawboxes
1685 \l@dsetupmaxlinecounts
1686 \l@dzeromaxlinecounts
1687 \l@dnumppstartsL=\z@
1688 \l@dnumppstartsR=\z@
1689 \l@dpscL=\z@
1690 \l@dpscR=\z@}
1691

```

## 25 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 1692 \newif\ifl@dusedbabel
\l@dusedbabeltrue 1693 \l@dusedbabelfalse

```

```

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

```

```

\l@dchecklang

```

```

\l@dbbl@set@language In babel the macro \bbl@set@language{<lang>} does the work when the language
<lang> is changed via \selectlanguage. Unfortunately for me, if it is given an

```

argument in the form of a control sequence it strips off the `\` character rather than expanding the command. I need a version that accepts an argument in the form `\lang` without it stripping the `\`.

```

1694 \newcommand*{\l@dbbl@set@language}[1]{%
1695   \edef\language{#1}%
1696   \select@language{\language}%
1697   \if@filesw
1698     \protected@write\@auxout{}\string\select@language{\language}%
1699     \addtocontents{toc}{\string\select@language{\language}%
1700     \addtocontents{lof}{\string\select@language{\language}%
1701     \addtocontents{lot}{\string\select@language{\language}%
1702   \fi}
1703

```

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

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

```

\theledlanguageR 1704 \providecommand{\selectlanguage}[1]{%
1705   \newcommand*{\l@duselanguage}[1]{%
1706     \gdef\theledlanguageL{}
1707     \gdef\theledlanguageR{}
1708

```

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

```

1709 \AtBeginDocument{%
1710   \@ifundefined{xpg@main@language}{%
1711     \@ifundefined{bbl@main@language}{%

```

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

```

1712   \l@dusedbabelfalse
1713   \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.

```

1714   \l@dusedbabeltrue
1715   \let\l@doldselectlanguage\selectlanguage
1716   \let\l@doldbbl@set@language\bbl@set@language
1717   \let\bbl@set@language\l@doldbbl@set@language
1718   \renewcommand{\selectlanguage}[1]{%
1719     \l@doldselectlanguage{#1}%
1720     \ifledRcol \gdef\theledlanguageR{#1}%
1721     \else      \gdef\theledlanguageL{#1}%
1722   \fi}

```

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

```
1723 \renewcommand*{\l@duselanguage}[1]{%
1724 \l@doldselectlanguage{#1}}
```

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

```
1725 \gdef\theledlanguageL{\bbl@main@language}%
1726 \gdef\theledlanguageR{\bbl@main@language}%
1727 }%
1728 }
```

If on Polyglossia

```
1729 { \let\old@otherlanguage\otherlanguage%
1730 \renewcommand{\otherlanguage}[2][]{%
1731 \selectlanguage[#1]{#2}%
1732 \ifledRcol \gdef\theledlanguageR{#2}%
1733 \else \gdef\theledlanguageL{#2}%
1734 \fi}%
1735 \let\l@duselanguage\select@language%
1736 \gdef\theledlanguageL{\xpg@main@language}%
1737 \gdef\theledlanguageR{\xpg@main@language}%
1738 }}
```

That's it.

```
\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue 1739 \newif\if@pstarts
\@pstartsfalse 1740 \newcommand*\check@pstarts}{%
\check@pstarts 1741 \@pstartsfalse
1742 \ifnum\l@dnumpstartsL>\l@dpscL
1743 \@pstartstrue
1744 \else
1745 \ifnum\l@dnumpstartsR>\l@dpscR
1746 \@pstartstrue
1747 \fi
1748 \fi
1749 }
1750
```

`\ifaraw@text` `\checkraw@text` checks whether the current Left or Right box is void or not. If `\araw@texttrue` one or other is not void it sets `\araw@texttrue`, otherwise both are void and it `\araw@textfalse` sets `\araw@textfalse`.

```
\checkraw@text 1751 \newif\ifaraw@text
1752 \araw@textfalse
1753 \newcommand*\checkraw@text}{%
1754 \araw@textfalse
1755 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}
1756 \araw@texttrue
1757 \else
1758 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}
```



```

1759     \araw@texttrue
1760     \fi
1761 \fi
1762 }
1763

```

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

```

1764 \newcommand*{\@writelinesinparL}{%
1765   \edef\next{%
1766     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1767   \next
1768   \global\@donereallinesL \z@}
1769 \newcommand*{\@writelinesinparR}{%
1770   \edef\next{%
1771     \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1772   \next
1773   \global\@donereallinesR \z@}
1774

```

## 26 Parallel columns

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

```

\@eledsectionR 1775 \newsavebox{\@eledsectionL}%
1776 \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.

```

1777 \newcommand*{\Columns}{%
1778   \eledsection@correcting@skip=-\baselineskip% Correction for sections' titles
1779   \setcounter{pstartL}{\value{pstartLold}}
1780   \setcounter{pstartR}{\value{pstartRold}}
1781   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1782     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1783   \fi

```

Start a group and zero counters, etc.

```

1784   \begingroup
1785     \l@dzeropenalties
1786     \endgraf\global\num@lines=\prevgraf
1787     \global\num@linesR=\prevgraf
1788     \global\par@line=\z@
1789     \global\par@lineR=\z@
1790     \global\l@dpscL=\z@
1791     \global\l@dpscR=\z@

```

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

```

1792     \check@pstarts
1793     \loop\if@pstarts
1794         \global\pstartnumtrue
1795         \global\pstartnumRtrue

```

Increment `\l@dpscL` and `\l@dpscR` which here count the numbers of left and right chunks.

```

1796     \global\advance\l@dpscL \@ne
1797     \global\advance\l@dpscR \@ne

```

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

```

1798     \checkraw@text
1799 {     \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.

```

1800         \l@duselanguage{\theledlanguageL}%
1801         \do@lineL
1802         \xifinlist{\the\l@dpscL}{\eled@sections@@}
1803             {%
1804                 \ifdefstring{\@eledsectmark}{L}%
1805                 {\csuse{eled@sectmark@the\l@dpscL}%
1806                 }{}}%
1807             \global\csundef{eled@sectmark@the\l@dpscL}%
1808             \savebox{\@eledsectionL}{\parbox[t]{}[t]{\Lcolwidth}{\vbox{} \print@eledse
1809             }%
1810             {}%
1811         \l@duselanguage{\theledlanguageR}%
1812         \do@lineR
1813         \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1814             {%
1815                 \ifdefstring{\@eledsectmark}{R}%
1816                 {\csuse{eled@sectmark@the\l@dpscR R}%
1817                 }{}}%
1818             \global\csundef{eled@sectmark@the\l@dpscR R}%
1819             \savebox{\@eledsectionR}{\parbox[t]{}[t]{\Rcolwidth}{\vbox{} \print@eledse
1820             }%
1821         \hb@xt@ \hsize{%
1822             \ifdefstring{\columns@position}{L}{\hfill }%
1823             \unhbox\l@dleftbox%
1824             \ifhbox\@eledsectionL%
1825                 \usebox{\@eledsectionL}%
1826             \fi%
1827             \print@columnseparator%
1828             \unhbox\l@drightbox%
1829             \ifhbox\@eledsectionR%
1830                 \usebox{\@eledsectionR}%

```

```

1831         \fi%
1832         \ifdefstring{\columns@position}{R}{\hfill}%
1833     }%
1834     \checkraw@text
1835     \checkverseL
1836     \checkverseR
1837     \checkpb@columns
1838     \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.

```

1839     \@writelinesinparL
1840     \@writelinesinparR
1841     \check@pstarts
1842     \ifbypstart@
1843         \write\linenum@out{\string\@set[1]}
1844         \resetprevline@
1845     \fi
1846     \ifbypstart@R
1847         \write\linenum@outR{\string\@set[1]}
1848         \resetprevline@
1849     \fi
1850     \addtocounter{pstartL}{1}
1851     \addtocounter{pstartR}{1}
1852     \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.

```

1853     \flush@notes
1854     \flush@notesR
1855     \endgroup
1856     \global\l@dpscL=\z@
1857     \global\l@dpscR=\z@
1858     \global\l@dnumpstartsL=\z@
1859     \global\l@dnumpstartsR=\z@
1860     \ignorespaces
1861     \global\instanzaLfalse
1862     \global\instanzaRfalse}
1863

```

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

```

1864 \def\print@columnseparator{%
1865     \ifdim\beforecolumnseparator<0pt%
1866         \hfill%
1867     \else%
1868         \hspace{\beforecolumnseparator}%
1869     \fi%

```

```

1870 \columnseparator%
1871 \ifdim\aftercolumnseparator<0pt%
1872   \hfill%
1873 \else%
1874   \hspace{\beforecolumnseparator}%
1875 \fi%
1876 }%
1877 %\end{macrocode}
1878 % \end{macro}
1879 % \begin{macro}{\checkpb@columns}
1880 % \cs{checkpb@columns} prevent or make pagebreaking in columns, depending of the use of \
1881 %   \begin{macrocode}
1882
1883 \newcommand{\checkpb@columns}{%
1884   \newif\if@pb
1885   \newif\if@nopb
1886   \IfStrEq{\led@pb@setting}{before}{%
1887     \numdef{\next@absline}{\the\absline@num+1}%
1888     \numdef{\next@abslineR}{\the\absline@numR+1}%
1889     \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
1890     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
1891       \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
1892       \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
1893         }{}
1894       \IfStrEq{\led@pb@setting}{after}{%
1895         \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
1896         \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
1897           \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
1898           \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
1899             }{}
1900 \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
1901 \if@pb\pagebreak[4]\fi
1902 }

```

`\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).

```

1903 \newcommand*{\columnseparator}{%
1904   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
1905 \newdimen\columnrulewidth
1906 \columnrulewidth=\z@
1907

```

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

```

1908 \newcommand*{\columnspostion}[1]{%
1909   \xdef\columns@position{#1}%
1910 }%

```

```

1911 \xdef\columns@position{R}%

\beforecolumnseparator \beforecolumnseparator and \aftercolumnseparator lengths are defined to
\aftercolumnseparator -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.

1912 \newlength{\beforecolumnseparator}%
1913 \setlength{\beforecolumnseparator}{-2pt}%
1914
1915 \newlength{\aftercolumnseparator}%
1916 \setlength{\aftercolumnseparator}{-2pt}%
1917

setwidthliketwocolumns@L The \setwidth... macros are called in \beginnumbering in a non-parallel
tpositionliketwocolumns@L typesetting context, to fix the width of the lines to be vertically aligned with par-
eositionliketwocolumns@L allel columns. They are also called at the beginning of a note's group, if some op-
setwidthliketwocolumns@C tions are enabled. The \setposition... macros are called in \beginnumbering
tpositionliketwocolumns@C in a non- parallel typesetting context to fix the position of the lines. The
eositionliketwocolumns@C \setnoteposition... macros are called in \xxxfootstart in a non- paral-
setwidthliketwocolumns@R lel typesetting context to fix the position of notes block.

tpositionliketwocolumns@R 1918 \newcommand{\setwidthliketwocolumns@L}{%
eositionliketwocolumns@R 1919 % Temporary dimension, initially equal to the standard hsize, i.e. text width
1920 % \begin{macrocode}
1921 \newdimen\temp%
1922 \temp=\hsize%

Hsize : Left + Right width
1923 \hsize=\Lcolwidth%
1924 \advance\hsize\Rcolwidth%

Now, calculating the remaining space
1925 \advance\temp-\hsize%

And multiply the hsize by 2/3 of this space
1926 \multiply\temp by 2%
1927 \divide\temp by 3%
1928 \advance\hsize\temp%
1929 }%
1930
1931 \newcommand{\setpositionliketwocolumns@L}{%
1932 \renewcommand{\ledrlfill}{\hfill}%
1933 }%
1934
1935 \newcommand{\setnotespositionliketwocolumns@L}{%
1936 }%
1937
1938

1939 \newcommand{\setwidthliketwocolumns@C}{%
1940 % Temporary dimension, initially equal to the standard hsize, i.e. text width

```

```

1941 \newdimen\temp%
1942 \temp=\hsize%
1943 % Hsize : Left + Right width
1944 \hsize=\Lcolwidth%
1945 \advance\hsize\Rcolwidth%
1946 % Now, calculating the remaining space
1947 \advance\temp-\hsize%

    And multiply the hsize by 1/2 of this space
1948 \divide\temp by 2%
1949 \advance\hsize\temp%
1950 }%
1951
1952 \newcommand{\setpositionliketwocolumns@C}{%
1953 \doinsidelinehook{\hfill}%
1954 \renewcommand{\ledrlfill}{\hfill}%
1955 }%
1956
1957 \newcommand{\setnotespositionliketwocolumns@C}{%
1958 \newdimen\temp%
1959 \newdimen\tempa%
1960 \temp=\hsize%
1961 \tempa=\Lcolwidth%
1962 \advance\tempa\Rcolwidth%
1963 \advance\temp-\tempa%
1964 \divide\temp by 2%
1965 \leftskip=\temp%
1966 \rightskip=-\temp%
1967 }%
1968
1969 \newcommand{\setwidthliketwocolumns@R}{%

    Temporary dimension, initially equal to the standard hsize, i.e. text width
1970 \newdimen\temp%
1971 \temp=\hsize%

    Hsize : Left + Right width
1972 \hsize=\Lcolwidth%
1973 \advance\hsize\Rcolwidth%

    Now, calculating the remaining space
1974 \advance\temp-\hsize%

    And multiply the hsize by 2/3 of this space
1975 \multiply\temp by 2%
1976 \divide\temp by 3%
1977 \advance\hsize\temp%
1978 }%
1979
1980 \newcommand{\setpositionliketwocolumns@R}{%
1981 \doinsidelinehook{\hfill}%

```

```

1982 }%
1983
1984 \newcommand{\setnotespositionliketwocolumns@R}{%
1985   \newdimen\temp%
1986   \newdimen\tempa%
1987   \temp=\hsize%
1988   \tempa=\Lcolwidth%
1989   \advance\tempa\Rcolwidth%
1990   \advance\temp-\tempa%
1991   \divide\temp by 2%
1992   \leftskip=\temp%
1993   \rightskip=-\temp%
1994 }%
1995

```

## 27 Parallel pages

This is considerably more complicated than parallel columns.

```

\numpagelinesL Counts for the number of lines on a left or right page, and the smaller of the
\numpagelinesR number of lines on a pair of facing pages.
\l@dminpagelines 1996 \newcount\numpagelinesL
                  1997 \newcount\numpagelinesR
                  1998 \newcount\l@dminpagelines
                  1999

```

**\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.

```

2000 \newcommand*{\Pages}{%
2001   \eledsection@correcting@skip=-2\baselineskip% line correcting for section titles.
2002   \setcounter{pstartL}{\value{pstartLold}}
2003   \setcounter{pstartR}{\value{pstartRold}}
2004   \parledgroup@notespacing@set@correction
2005   \typeout{}
2006   \typeout{***** PAGES *****}
2007   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
2008     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2009   \fi

```

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

```

2010 \cleartol@devenpage
2011 \begingroup
2012   \l@dzeropenalties
2013   \endgraf\global\num@lines=\prevgraf
2014   \global\num@linesR=\prevgraf
2015   \global\par@line=\z@
2016   \global\par@lineR=\z@

```

```

2017    \global\l@dpscL=\z@
2018    \global\l@dpscR=\z@
2019    \writtenlinesLfalse
2020    \writtenlinesRfalse

```

Check if there are chunks to be processed.

```

2021    \check@pstarts
2022    \loop\if@pstarts

```

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

```

2023    \global\advance\l@dpscL \@ne
2024    \global\advance\l@dpscR \@ne

```

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

```

2025    \getlinesfromparlistL
2026    \getlinesfromparlistR
2027    \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2028    {\usernamecount{l@dmmaxlinesinpar\the\l@dpscL}}%
2029    \check@pstarts
2030    \repeat

```

Zero the counts again, ready for the next bit.

```

2031    \global\l@dpscL=\z@
2032    \global\l@dpscR=\z@

```

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

```

2033    \getlinesfrompagelistL
2034    \getlinesfrompagelistR
2035    \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2036    {\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.

```

2037    \check@pstarts
2038    \if@pstarts

```

Increment the chunk counts to get the first pair.

```

2039    \global\advance\l@dpscL \@ne
2040    \global\advance\l@dpscR \@ne

```

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

```

2041    \global\@donereallinesL=\z@
2042    \global\@donetotallinesL=\z@
2043    \global\@donereallinesR=\z@
2044    \global\@donetotallinesR=\z@

```

Start a loop over the boxes (chunks).

```

2045    \checkraw@text

```



```

2046 %      \begingroup
2047 {      \loop\ifaraw@text

```

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

```

2048      \checkpageL
2049      \l@duselanguage{\theledlanguageL}%
2050 %%%      \begingroup
2051 {      \loop\ifl@dsamepage%

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

2052      \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
2053      \csuse{before@pstartL@the\l@dpscl}%
2054      \global\csundef{before@pstartL@the\l@dpscl}
2055      \do@lineL
2056      \xifinlist{\the\l@dpscl}{\eled@sections@@}
2057      {\print@eledsectionL}%
2058      {%}
2059      \advance\numpagelinesL \@ne
2060      \ifshiftedpstarts
2061          \ifdim\ht\l@dleftbox>0pt\hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}\fi%
2062      \else%
2063          \parledgroup@correction@notespacing{L}
2064          \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2065      \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.

```

2066      \get@nextboxL%
2067      \ifprint@last@after@pendL%
2068      \csuse{after@pendL@the\l@dpscl}%
2069      \global\csundef{after@pendL@the\l@dpscl}%
2070      \fi%
2071      \checkpageL%
2072      \checkverseL
2073      \checkpbl
2074      \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.

```

2075      \ifl@pagefull
2076      \@writelinesonpageL{\the\numpagelinesL}%
2077      \else
2078      \@writelinesonpageL{1000}%
2079      \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.

```

2080      \numpagelinesL \z@
2081      \parledgroup@correction@notespacing@init
2082      \clearl@dleftpage }%

```

Now do the same for the right text.

```

2083      \checkpageR
2084      \l@duselanguage{\theledlanguageR}%
2085 {
2086      \loop\ifl@dsamepage%
2087      \initnumbering@sectcountR
2088      \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{%
2089      \csuse{before@pstartR@the\l@dpscR}%
2090      \global\csundef{before@pstartR@the\l@dpscR}
2091      \do@lineR
2092      \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
2093      {\print@eledsectionR}%
2094      }{}%
2095      \advance\numpagelinesR \@ne
2096      \ifshiftedpstarts
2097      \ifdim\ht\l@drightbox>0pt\hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2098      \else%
2099      \parledgroup@correction@notespacing{R}
2100      \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2101      \fi
2102      \get@nextboxR%
2103      \ifprint@last@after@pendR%
2104      \csuse{after@pendR@the\l@dpscR}%
2105      \global\csundef{after@pendR@the\l@dpscR}%
2106      \fi%
2107      \checkpageR%
2108      \checkverseR
2109      \checkpbR
2110      \repeat
2111      \ifl@dpagfull
2112      \@writelinesonpageR{\the\numpagelinesR}%
2113      \else
2114      \@writelinesonpageR{1000}%
2115      \fi
2116      \numpagelinesR=\z@
2117      \parledgroup@correction@notespacing@init

```

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

```

2117      \clearl@drightpage}

```

More to do? If there is we have to get the number of lines for the next pair of pages before starting to output them.

```

2118      \checkraw@text

```

```

2119      \ifaraw@text
2120      \getlinesfrompagelistL
2121      \getlinesfrompagelistR
2122      \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2123      {\l@dminpagelines}%
2124      \fi
2125      \repeat}

```

We have now output the text from all the chunks.

```

2126      \fi
      Make sure that there are no inserts hanging around.
2127      \flush@notes
2128      \flush@notesR
2129      \endgroup

```

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

```

2130      \global\l@dpscL=\z@
2131      \global\l@dpscR=\z@
2132      \global\l@dnumpestartsL=\z@
2133      \global\l@dnumpestartsR=\z@
2134      \global\instanzaLfalse
2135      \global\instanzaRfalse
2136      \ignorespaces}
2137

```

`\ledstrutL` Struts inserted into left and right text lines.

```

\ledstrutR 2138 \newcommand*{\ledstrutL}{\strut}
2139 \newcommand*{\ledstrutR}{\strut}
2140

```

`\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. `\clearl@dleftpage` and `\clearl@drightrightpage` get us onto an odd and even page, respectively, checking that we end up on the immediately next page.

```

2141 \providecommand{\cleartoevenpage}[1][\@empty]{%
2142   \clearpage
2143   \ifodd\c@page\hbox{#1}\clearpage\fi}
2144 \newcommand*{\cleartol@devenpage}{%
2145   \ifdim\pagetotal<\topskip% on an empty page
2146   \else
2147   \clearpage
2148   \fi
2149   \ifodd\c@page\hbox{#1}\clearpage\fi}
2150 \newcommand*{\clearl@dleftpage}{%
2151   \clearpage
2152   \ifodd\c@page\else
2153   \led@err@LeftOnRightPage

```

```

2154 \hbox{}%
2155 \cleardoublepage
2156 \fi}
2157 \newcommand*{\clearl@drightpage}{%
2158 \clearpage
2159 \ifodd\c@page
2160 \led@err@RightOnLeftPage
2161 \hbox{}%
2162 \cleartoevenpage
2163 \fi}
2164

```

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

```

\cs@linesinparR 2165 \newcommand*{\getlinesfromparlistL}{%
2166 \ifx\linesinpar@listL\empty
2167 \gdef\cs@linesinparL{0}%
2168 \else
2169 \gl@p\linesinpar@listL\to\cs@linesinparL
2170 \fi}
2171 \newcommand*{\getlinesfromparlistR}{%
2172 \ifx\linesinpar@listR\empty
2173 \gdef\cs@linesinparR{0}%
2174 \else
2175 \gl@p\linesinpar@listR\to\cs@linesinparR
2176 \fi}
2177

```

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

```

\cs@linesonpageR 2178 \newcommand*{\getlinesfrompagelistL}{%
2179 \ifx\linesonpage@listL\empty
2180 \gdef\cs@linesonpageL{1000}%
2181 \else
2182 \gl@p\linesonpage@listL\to\cs@linesonpageL
2183 \fi}
2184 \newcommand*{\getlinesfrompagelistR}{%
2185 \ifx\linesonpage@listR\empty
2186 \gdef\cs@linesonpageR{1000}%
2187 \else
2188 \gl@p\linesonpage@listR\to\cs@linesonpageR
2189 \fi}
2190

```

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

```

2191 \newcommand*{\@writelinesonpageL}[1]{%

```

```

2192 \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2193 \next}
2194 \newcommand*{\@writelinesonpageR}[1]{%
2195 \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2196 \next}
2197

\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>.

2198 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2199 \ifnum #2>#1\relax
2200 #3=#2\relax
2201 \else
2202 #3=#1\relax
2203 \fi}
2204 \newcommand*{\l@dcalc@minoftwo}[3]{%
2205 \ifnum #2<#1\relax
2206 #3=#2\relax
2207 \else
2208 #3=#1\relax
2209 \fi}
2210

\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@dpagetruefull is set FALSE and
\l@dsamepagefalse \ifl@dsamepage is set TRUE. If the page is spatially full then \ifl@dpagetruefull
\ifl@dpagetruefull is set TRUE and \ifl@dsamepage is set FALSE. If it is not spatially full but
\l@dpagetruefulltrue the maximum number of lines have been output then both \ifl@dpagetruefull and
\l@dpagetruefullfalse \ifl@dsamepage are set FALSE.

\checkpageL 2211 \newif\ifl@dsamepage
\checkpageR 2212 \l@dsamepagetrue
2213 \newif\ifl@dpagetruefull
2214
2215 \newcommand*{\checkpageL}{%
2216 \l@dpagetruefulltrue
2217 \l@dsamepagetrue
2218 \check@goal
2219 \ifdim\pagetotal<\ledthegoal
2220 \ifnum\numpagelinesL<\l@dminpagelines
2221 \else
2222 \l@dsamepagefalse
2223 \l@dpagetruefullfalse
2224 \fi
2225 \else
2226 \l@dsamepagefalse
2227 \l@dpagetruefulltrue
2228 \fi%

```

```

2229 \ifprint@last@after@pendL%
2230     \l@dpagfullfalse%
2231     \l@dsamepagefalse%
2232     \print@last@after@pendLfalse%
2233 \fi%
2234 }%
2235
2236 \newcommand*{\checkpageR}{%
2237     \l@dpagfulltrue
2238     \l@dsamepagetrue
2239     \check@goal
2240     \ifdim\pagetotal<\ledthegoal
2241         \ifnum\numpagelinesR<\l@dminpagelines
2242             \else
2243                 \l@dsamepagefalse
2244                 \l@dpagfullfalse
2245             \fi
2246         \else
2247             \l@dsamepagefalse
2248             \l@dpagfulltrue
2249         \fi%
2250     \ifprint@last@after@pendR%
2251         \l@dpagfullfalse%
2252         \l@dsamepagefalse%
2253         \print@last@after@pendRfalse%
2254     \fi%
2255 }%
2256

```

\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.

```

2257 \newcommand{\checkpbL}{
2258     \IfStrEq{\led@pb@setting}{after}{
2259         \xifinlistcs{\the\absline@num}{\l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{
2260             \xifinlistcs{\the\absline@num}{\l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{
2261                 {}
2262             \IfStrEq{\led@pb@setting}{before}{
2263                 \numdef{\next@absline}{\the\absline@num+1}
2264                 \xifinlistcs{\next@absline}{\l@prev@pb}{\l@dpagfulltrue\l@dsamepagefalse}{
2265                     \xifinlistcs{\next@absline}{\l@prev@nopb}{\l@dpagfullfalse\l@dsamepagetrue}{
2266                         {}
2267                     }
2268                 }
2269 \newcommand{\checkpbR}{
2270     \IfStrEq{\led@pb@setting}{after}{
2271         \xifinlistcs{\the\absline@numR}{\l@prev@pbR}{\l@dpagfulltrue\l@dsamepagefalse}{
2272             \xifinlistcs{\the\absline@numR}{\l@prev@nopbR}{\l@dpagfullfalse\l@dsamepagetrue}{
2273                 {}
2274             \IfStrEq{\led@pb@setting}{before}{

```

```

2280 \newcommand{\checkverseL}{%
2281 \ifinstanzaL
2282 \iflednopbinverse
2283 \ifinserthangingsymbol
2284 \numgdef{\prev@abslineverse}{\the\absline@num-1}
2285 \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{}
2286 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\prev@abslineverse}\fi}{%
2287 \fi
2288 \fi
2289 \fi
2290 }
2291 \newcommand{\checkverseR}{%
2292 \ifinstanzaR
2293 \iflednopbinverse
2294 \ifinserthangingsymbolR
2295 \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2296 \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{}
2297 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\prev@abslineverse}\fi}{%
2298 \fi
2299 \fi
2300 \fi
2301 }

```

```

2302 \newdimen\ledthegoal
2303 \ifshiftedpstarts
2304     \newcommand*{\goalfraction}{0.95}
2305 \else
2306     \newcommand*{\goalfraction}{0.9}
2307 \fi
2308
2309 \newcommand*{\check@goal}{%
2310     \ledthegoal=\goalfraction\pagegoal}
2311

```

```

\ifwrittenlinesL 2312 \newif\ifwrittenlinesL
                  2313 \newif\ifwrittenlinesR
                  2314

```

`\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.

```

2315 \newcommand*{\get@nextboxL}{%
2316   \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}% box is not empty

   The current box is not empty; do nothing.
2317   \else%                                     box is empty

   The box is empty. Check if enough lines (real and blank) have been output.
2318     \ifnum\usernamecount{1@dmaxlinesinpar\the\1@dpscL}>\@donetotallinesL
2319       \parledgroup@notes@endL
2320     \else

   Sufficient lines have been output.
2321       \ifnum\usernamecount{1@dmaxlinesinpar\the\1@dpscL}=\@donetotallinesL
2322         \parledgroup@notes@endL
2323       \fi
2324       \ifwrittenlinesL\else

   Write out the number of lines done, and set the boolean so this is only done once.
2325         \@writelinesinparL
2326         \writtenlinesLtrue
2327       \fi
2328       \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. Increment the pstartL counter.
2329         \writtenlinesLfalse
2330         \ifbypstart@
2331           \ifnum\value{pstartL}<\value{pstartLold}
2332           \else
2333             \global\line@num=0
2334             \resetprevline@
2335           \fi
2336         \fi

2337 % Add the content of the optional argument of the previous \cs{pend}.
2338 %   \begin{macrocode}
2339       \csuse{after@pendL@the\1@dpscL}%
2340       \global\csundef{after@pendL@the\1@dpscL}%
2341 %   \end{macrocode}
2342 %   \begin{macrocode}
2343       \addtocounter{pstartL}{1}
2344       \global\pstartnumtrue
2345       \1@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\1@dpscL}}%
2346                           {\the\@donetotallinesL}%
2347                           {\usernamecount{1@dmaxlinesinpar\the\1@dpscL}}%
2348       \global\@donetotallinesL \z@
2349       \global\advance\1@dpscL \@ne

```

Add notes of parallel ledgroup.



```

2350      \parledgroup@notes@endL
2351      \parledgroup@correction@notes@spacing@final{L}
2352      \else
2353 % Add the content of the optional argument of the last \cs{pend}.
2354 %      \begin{macrocode}
2355      \print@last@after@pendLtrue%
2356 %      \end{macrocode}
2357 %      \begin{macrocode}
2358      \fi
2359      \fi
2360      \fi}

2361 \newcommand*{\get@nextboxR}{%
2362   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2363   \else% box is empty
2364     \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2365       \parledgroup@notes@endR
2366     \else
2367       \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2368         \parledgroup@notes@endR
2369       \fi
2370       \ifwrittenlinesR\else
2371         \@writelinesinparR
2372         \writtenlinesRtrue
2373       \fi
2374       \ifnum\l@dnumpstartsR>\l@dpscR
2375         \writtenlinesRfalse
2376         \ifbypstartR
2377           \ifnum\value{pstartR}<\value{pstartRold}
2378             \else
2379               \global\line@numR=0
2380               \resetprevline@
2381             \fi
2382           \fi
2383           \csuse{after@pendR@the\l@dpscR}%
2384           \global\csundef{after@pendR@the\l@dpscR}%
2385           \addtocounter{pstartR}{1}
2386           \global\pstartnumRtrue
2387           \l@dcalc@maxoftwo{\the\usenamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2388             {\the\@donetotallinesR}%
2389             {\usenamecount{1@dmaxlinesinpar\the\l@dpscR}}}%
2390           \global\@donetotallinesR \z@
2391           \global\advance\l@dpscR \@ne
2392           \parledgroup@notes@endR
2393           \parledgroup@correction@notes@spacing@final{R}
2394         \else
2395           \print@last@after@pendRtrue%
2396         \fi
2397       \fi
2398     \fi}

```

2399

## 28 Sections' titles' commands

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

```
2400 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2401 \eledsectnotoc{R}
```

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

```
2402 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2403 \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`.

```
2404 \newskip\eledsection@correcting@skip
```

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

```
2405 \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`.

`\l@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).

```
2406 \def\l@prev@pbR{}
2407 \def\l@prev@nopbR{}
```

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

```
2408 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2409 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2410 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2411 \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.

```
2412 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2413 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{\#1}}
2414 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2415 \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.  
`\parledgroup@type@` `\parledgroupseries` contains the type of the footnote: critical (Xfootnote) or  
familiar (footnoteX).

```
2416 \newmarks\parledgroup@
2417 \newmarks\parledgroup@series
2418 \newmarks\parledgroup@type
```

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

```
2419 \newcommand{\parledgroup@notes@startL}{%
2420   \ifnum\usenamecount{\l@maxlinesinpar\the\l@dpscL}>0%
2421     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmarks\parledg
2422     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmarks\parledg
2423     \fi%
2424     \global\ledgroupnotesL@true%
2425     \insert@noterule@ledgroup{L}%
2426 }
2427 \newcommand{\parledgroup@notes@startR}{%
2428   \ifnum\usenamecount{\l@maxlinesinpar\the\l@dpscR}>0%
2429     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{bhooknoteX@\splitfirstmarks\parledg
2430     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{bhookXnote@\splitfirstmarks\parledg
2431     \fi%
2432     \global\ledgroupnotesR@true%
2433     \insert@noterule@ledgroup{R}%
2434 }
```

`\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.

```
2435 \newcommand{\parledgroup@notes@endL}{%
2436   \global\ledgroupnotesL@false%
2437 }
2438 \newcommand{\parledgroup@notes@endR}{%
2439   \global\ledgroupnotesR@false%
2440 }
```

`\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`

```

2441 \newcommand{\insert@noterule@ledgroup}[1]{
2442   \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2443     \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2444       \csuse{ifledgroupnotes#1@}
2445       \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2446       \csuse{\splitbotmarks\parledgroup@series footnoterule}
2447       \fi
2448     }
2449   {}
2450   \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2451     \csuse{ifledgroupnotes#1@}
2452     \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2453     \csuse{footnoterule\splitbotmarks\parledgroup@series}
2454     \fi
2455   }{}
2456 }
2457 {}
2458 }
```

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

```

2459 \newcommand{\parledgroupnotespacing}{}

```

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

```

2460 \dimdef{\parledgroup@notespacing@correction}{0pt}
2461 \newcommand{\parledgroup@notespacing@set@correction}{%
2462   {\notefontsetup\parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
2463   \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing}%
2464 }

```

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

```

2465 \newcommand{\parledgroup@correction@notespacing@init}{
2466   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2467   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2468 }
2469 \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`.

```

2470 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2471   \ifparledgroup
2472   \vspace{\parledgroup@notespacing@correction@accumulated}
2473   \parledgroup@correction@notespacing@init%
2474   \ifstrequal{#1}{L}{
2475     \numdef{\@checking}{\the\l@dpscL-1}
2476   }{
2477     \numdef{\@checking}{\the\l@dpscR-1}
2478   }
2479   \dimdef{\@beforenotes@current@diff}{\csuse{@parledgroup@beforenotes@\@checking L}-\csuse{@parledg
2480   \ifstrequal{#1}{L}%
2481     {% Left
2482     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\@beforenotes@current@diff}}%
2483     }%
2484     {% Right
2485     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\@beforenotes@current@diff}}{}}
2486     }%
2487   \fi
2488 }

```

`\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.

```

2489 {}
2490 \newcommand{\parledgroup@correction@notespacing}[1]{%
2491   \csuse{ifledgroupnotes#1@}%
2492   \vspace{-\parledgroup@notespacing@correction}%
2493   \dimdef{\parledgroup@notespacing@correction@accumulated}{\parledgroup@notespacing@correction@ac
2494   \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo+
2495   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip}{\advance\numpagelinesL
2496   \dimdef{\parledgroup@notespacing@correction@modulo}{\parledgroup@notespacing@correction@modulo-
2497   }% mean greater than equal
2498   \fi%
2499 }

```

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

```

2500 \dimdef\parledgroup@beforenotesL{0pt}
2501 \dimdef\parledgroup@beforenotesR{0pt}

```

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

```

2502 \newcommand{\parledgroup@beforenotes@save}[1]{
2503   \ifparledgroup
2504     \csdingdef{@parledgroup@beforenotes@\the\csuse{1@dnumstarts#1}#1}{\csuse{parledgroup
2505       \csdingdef{parledgroup@beforenotes#1}{Opt}
2506     \fi
2507 }
```

## 31 The End

i/code<sub>i</sub>

## Appendix A Some things to do when changing version

### Appendix A.1 Migration to eledpar 1.4.3

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

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

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

See the two following examples:

With standard `\hangingsymbol`:

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

With the modification of `\hangingsymbol`:

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

## References

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

## Index

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

<b>Symbols</b>	<code>\@adv</code> . . . . .	370, 637, 638
<code>\&amp;</code> . . . . .	1590, 1591, 1595, 1612, 1633	<code>\@afterindentfalse</code> . . . . . 773
<code>\@M</code> . . . . .	1601	<code>\@arabic</code> . . . . . 222, 223, 822, 825

- \@astanza@line . . . . . 1611, 1617, 1620
  - \@auxout . . . . . 1424, 1436, 1698
  - \@beforenotes@current@diff . . . . .  
     . . . . . 2479, 2482, 2485
  - \@chapter . . . . . 774
  - \@checking . . . . . 2475, 2477, 2479
  - \@cs@linesinparL . . . . . 2027, 2165
  - \@cs@linesinparR . . . . . 2027, 2165
  - \@cs@linesonpageL . . . . . 2035, 2122, 2178
  - \@cs@linesonpageR . . . . . 2035, 2122, 2178
  - \@currentlabel . . . . . 865, 908
  - \@donereallinesL . . . . .  
     . . . . . 967, 996, 1766, 1768, 2041
  - \@donereallinesR . . . . .  
     . . . . . 967, 1068, 1771, 1773, 2043
  - \@donetotallinesL . . . . . 967, 997,  
     1000, 2042, 2318, 2321, 2346, 2348
  - \@donetotallinesR . . . . . 967, 1069,  
     1072, 2044, 2364, 2367, 2388, 2390
  - \@edtext@false . . . . . 695, 721
  - \@edtext@true . . . . . 673, 699
  - \@eled@sectioningfalse . . . . . 1020
  - \@eled@sectioningtrue . . . . . 1018
  - \@eledsectionL . . . . . 1775, 1808, 1824, 1825
  - \@eledsectionR . . . . . 1775, 1819, 1829, 1830
  - \@eledsectmark . . . . . 1015, 1804, 1815, 2402
  - \@eledsectnotoc . . . . . 1014, 2052, 2087, 2400
  - \@gobble . . . . . 559
  - \@gobbletwo . . . . . 564
  - \@insertR . . . . . 1374–1376, 1389–1391
  - \@l@dttempcnta . . . . . 443,  
     445, 447, 448, 452, 454, 456,  
     457, 1126, 1165, 1166, 1168,  
     1170, 1173, 1174, 1191–1195,  
     1197, 1204, 1209, 1213, 1221,  
     1226, 1230, 1263, 1266, 1268, 1272
  - \@l@dttempcntb . . . . . 178, 180, 182,  
     1156, 1157, 1204, 1209, 1213,  
     1221, 1226, 1230, 1255, 1259,  
     1272, 1280–1282, 1284, 1305–  
     1307, 1309, 1326–1328, 1330,  
     1465, 1466, 1494–1496, 1498,  
     1661–1665, 1669–1672, 1676–1679
  - \@lab . . . . . 563, 1415, 1427, 1452
  - \@lemmacommand@false . . . . . 696, 722
  - \@lock . . . . . 984, 1106
  - \@lockR . . . . . 56, 314, 316, 318, 331, 477,  
     493, 494, 496, 497, 525, 526,  
     528, 1055, 1088, 1132, 1134,  
     1135, 1137, 1218, 1235, 1237, 1239
  - \@lopL . . . . . 588, 2192
  - \@lopR . . . . . 559, 588, 2195
  - \@nl . . . . . 292, 620, 622
  - \@nl@reg . . . . . 341
  - \@nl@regR . . . . . 292
  - \@nobreakfalse . . . . . 831, 875, 1017
  - \@nobreaktrue . . . . . 829, 833, 873, 877, 1017
  - \@noprtrue . . . . . 1891, 1892, 1897, 1898
  - \@oldnobreak . . . . . 829, 831, 873, 875, 926, 947
  - \@pbtrue . . . . . 1889, 1890, 1895, 1896
  - \@pend . . . . . 579, 1766
  - \@pendR . . . . . 579, 1771
  - \@pstartsfalse . . . . . 1739
  - \@pstartstrue . . . . . 1739
  - \@ref . . . . . 549
  - \@ref@reg . . . . . 577
  - \@schapter . . . . . 774
  - \@set . . . . . 402, 644, 645, 1843, 1847
  - \@startstanza . . . . . 786, 787, 809, 810
  - \@sw . . . . . 564
  - \@tag . . . . . 675, 701, 1519, 1521, 1525
  - \@templ@d . . . . . 1484, 1486
  - \@templ@n . . . . . 1485, 1486
  - \@writelinesinparL . . . . . 1764, 1839, 2325
  - \@writelinesinparR . . . . . 1764, 1840, 2371
  - \@writelinesonpageL . . . . . 2076, 2078, 2191
  - \@writelinesonpageR . . . . . 2111, 2113, 2191
  - \@xloop . . . . . 1387
- A**
- \absline@num . . . . .  
     366, 436, 450, 469, 1097, 1887,  
     1895, 1897, 2259, 2260, 2263, 2284
  - \absline@numR . . . . . 51, 239, 294,  
     297, 300, 433, 441, 462, 481,  
     515, 543, 554, 1079, 1118, 1119,  
     1156, 1373, 1888, 1896, 1898,  
     2271, 2272, 2275, 2295, 2412, 2414
  - \actionlines@list . . . . .  
     . . . . . 284, 287, 436, 450, 469
  - \actionlines@listR . . . . .  
     . . . . . 243, 260, 276, 279, 433,  
     441, 462, 481, 515, 543, 1178, 1181
  - \actions@list . . . . . 288, 437, 457, 471, 473
  - \actions@listR . . . . .  
     . . . . . 243, 261, 280, 434, 448, 464,  
     466, 483, 492, 517, 524, 544, 1182
  - \add@inserts . . . . . 993, 1007
  - \add@inserts@nextR . . . . . 1362
  - \add@insertsR . . . . . 1064, 1362



- \add@penaltiesL ..... 995, 1383
  - \add@penaltiesR ..... 1067, 1383
  - \addtocontents ..... 1699–1701
  - \addtocounter ..... 928,  
949, 1013, 1850, 1851, 2343, 2385
  - \advancelabel@refs ..... 1422, 1434
  - \advanceline ..... 636, 667
  - \affixline@num ..... 991
  - \affixline@numR ..... 1062, 1188
  - \affixpstart@numL ..... 1005, 1295
  - \affixpstart@numR ..... 1295
  - \affixside@note ..... 993, 1007
  - \affixside@noteR ..... 1064, 1471
  - \aftercolumnseparator . . 6, 1871, 1912
  - \appto ..... 1477, 1478
  - \araw@textfalse ..... 1751
  - \araw@texttrue ..... 1751
  - astanza (environment) ..... 10, 1593
  - \at@begin@pairs ..... 750, 755, 756
  - \at@every@pend ..... 932, 953
  - \at@every@pstart ..... 868, 911
  - \AtBeginDocument . . . 1448, 1683, 1709
  - \AtBeginPairs ..... 5, 755
- B**
- \ballast@count ..... 1116, 1121
  - \bbl@main@language ..... 1725, 1726
  - \bbl@set@language ..... 1716, 1717
  - \beforecolumnseparator .....  
..... 6, 1865, 1868, 1874, 1912
  - \beginnumbering ..... 9, 797, 836
  - \beginnumberingR . . . 42, 123, 797, 880
  - \bfseries ..... 822, 825
  - \bypage@Rfalse ..... 143, 158, 163
  - \bypage@Rtrue ..... 143, 153
  - \bypstart@Rfalse ..... 143, 154, 164
  - \bypstart@Rtrue ..... 143, 159
- C**
- \c@ballast ..... 1121
  - \c@chapter ..... 104
  - \c@chapterR ..... 104
  - \c@firstlinenumR ..... 187, 1261
  - \c@firstsublinenumR ..... 191, 1256
  - \c@linenumincrementR ..... 187, 1261
  - \c@page 620, 622, 2143, 2149, 2152, 2159
  - \c@pstart ..... 1437
  - \c@pstartL ..... 822
  - \c@pstartR ..... 825, 1425
  - \c@section ..... 105
  - \c@sectionR ..... 105
  - \c@sublinenumincrementR . . 191, 1256
  - \c@subsection ..... 106
  - \c@subsectionR ..... 106
  - \c@subsubsection ..... 107
  - \c@subsubsectionR ..... 107
  - \ch@ck@l@ckR ..... 1188
  - \ch@cksub@l@ckR ..... 1188
  - \ch@cksub@lockR ..... 1257
  - \chapter ..... 759, 760, 769
  - \chapterinpages ..... 746, 760, 771
  - \chardef ..... 1590
  - \check@goal ..... 2218, 2239, 2302
  - \check@pstarts .....  
..... 1739, 1792, 1841, 2021, 2029, 2037
  - \checkpageL ..... 2048, 2071, 2211
  - \checkpageR ..... 2083, 2106, 2211
  - \checkpb@columns . . . 1837, 1879, 1883
  - \checkpbL ..... 2073, 2257
  - \checkpbR ..... 2108, 2257
  - \checkraw@text .....  
..... 1751, 1798, 1834, 2045, 2118
  - \checkverseL ..... 1835, 2072, 2280
  - \checkverseR ..... 1836, 2107, 2280
  - \cleardoublepage ..... 2155
  - \clearl@dleftpage ..... 2082, 2141
  - \clearl@drightpage ..... 2117, 2141
  - \cleartoevenpage ..... 2141
  - \cleartol@devenpage ..... 2010, 2141
  - \closeout ..... 95, 601, 607, 610, 614
  - \columnrulewidth ..... 6, 1903
  - \Columns ..... 5, 1777
  - \columns@position . . 1822, 1832, 1908
  - \columnseparator ..... 6, 1870, 1903
  - \columnsposition ..... 6, 1908
  - \correcthangingL ..... 1009, 1571
  - \correcthangingR ..... 1571
  - \countLline ..... 962, 973
  - \countRline ..... 962, 1044
  - \critext ..... 672
  - \cs ..... 1880, 2337, 2353
  - \csdimgdef ..... 2504, 2505
  - \csgdef ..... 868,  
869, 911, 912, 932, 933, 953, 954
  - \csundef ..... 1021, 1807, 1818,  
2054, 2069, 2089, 2104, 2340, 2384
  - \csuse ..... 1019,  
1540, 1551, 1805, 1816, 2053,  
2068, 2088, 2103, 2339, 2383,

2421, 2422, 2429, 2430, 2444–  
2446, 2451–2453, 2479, 2491, 2504

**D**

`\DeclareOption` ..... 8–11  
`\def@tempb` ..... 161  
`\dimdef` ..... 2460, 2466, 2467,  
2479, 2493, 2494, 2496, 2500, 2501  
`\dimen` ..... 623, 624, 628–630, 634  
`\dingdef` ..... 2462, 2463  
`\divide` 1193, 1927, 1948, 1964, 1976, 1991  
`\do@actions` ..... 1098  
`\do@actions@fixedcodeR` ..... 1125  
`\do@actions@nextR` ..... 1125  
`\do@actionsR` ..... 1080, 1125  
`\do@ballast` ..... 1099  
`\do@ballastR` ..... 1081, 1116  
`\do@insidelineLhook` . 1009, 1034, 1037  
`\do@insidelineRhook` ..... 1035, 1037  
`\do@lineL` ..... 972, 1801, 2055  
`\do@lineLhook` ..... 977, 1032, 1037  
`\do@lineR` ..... 1042, 1812, 2090  
`\do@lineRhook` ..... 1033, 1037, 1048  
`\do@lockoff` ..... 512  
`\do@lockoffL` ..... 536  
`\do@lockoffR` ..... 512  
`\do@lockon` ..... 477  
`\do@lockonL` ..... 509  
`\do@lockonR` ..... 477  
`\doinsidelinehook` ..... 1953, 1981  
`\doinsidelineLhook` ..... 1032  
`\doinsidelineRhook` ..... 1032  
`\dolineLhook` ..... 1032  
`\dolineRhook` ..... 1032  
`\dolistloop` ..... 1482, 1502, 1509  
`\dummy@ref` ..... 558

**E**

`\edfont@info` ..... 729, 732, 738, 741  
`\edlabel` ..... 1409  
`\edtext` ..... 698  
`\eled@sectioningR@out` .. 70, 95, 2405  
`\eled@sections@@` 992, 1017, 1802, 2056  
`\eled@sectionsR@@` 67, 1063, 1813, 2091  
`\eledpar@error` ..... 21, 30, 33, 35  
`\eledsection@correcting@skip` ..  
..... 1029, 1778, 2001, 2404  
`\eledsectmark` ..... 14, 2402  
`\eledsectnotoc` ..... 14, 2400

`\empty` .. 80, 83, 276, 284, 686, 712,  
727, 736, 845, 889, 1178, 1260,  
1268, 1364–1366, 1377, 1388,  
1416, 1428, 2166, 2172, 2179, 2185  
`\end@lemmas` ..... 686, 687, 712, 713  
`\endashchar` ..... 1399  
`\endgraf` ..... 922, 943, 1786, 2013  
`\endline@num` ..... 567, 573  
`\endlock` ..... 656, 1599, 1608, 1613  
`\endnumbering` ..... 9, 73, 127, 798  
`\endnumberingR` 45, 73, 110, 122, 135, 798  
`\endpage@num` ..... 566, 573  
`\endstanzaextra` ..... 1615  
`\endsub` ..... 623  
`\endsubline@num` ..... 568, 574  
`\enlargethispage` ..... 1900  
environments:  
    `astanza` ..... 10, 1593  
    `Leftside` ..... 7, 778  
    `pages` ..... 6, 746  
    `pairs` ..... 5, 746  
    `Rightside` ..... 7, 795  
`\expandonce` .... 1521, 1525, 1541, 1552  
`\extensionchars` .... 62, 116, 132, 140

**F**

`\f@x@l@cksR` ..... 1188  
`\first@linenum@out@Rfalse` .. 596, 602  
`\first@linenum@out@Rtrue` ..... 596  
`\firstlinenum` ..... 7, 196  
`\firstlinenum*` ..... 7, 196  
`\firstsublinenum` ..... 7, 196  
`\firstsublinenum*` ..... 7, 196  
`\fix@page` ..... 337, 344  
`\flag@end` .....  
..... 623, 682, 692, 693, 708, 718, 719  
`\flag@start` .....  
..... 623, 681, 682, 693, 707, 708, 719  
`\flush@notes` ..... 1853, 2127  
`\flush@notesR` ..... 1386, 1854, 2128  
`\footnotelang@lua` ..... 1535, 1546  
`\footnotelang@poly` ..... 1539, 1550  
`\fullstop` . 235, 1396, 1398, 1400, 1402

**G**

`\get@linelistfile` ..... 272  
`\get@nextboxL` ..... 2066, 2315  
`\get@nextboxR` ..... 2101, 2315  
`\getline@numL` ..... 983, 1096  
`\getline@numR` ..... 1054, 1078

- `\getlinesfrompagelistL` ..... 2033, 2120, 2178  
`\getlinesfrompagelistR` ..... 2034, 2121, 2178  
`\getlinesfromparlistL` ... 2025, 2165  
`\getlinesfromparlistR` ... 2026, 2165  
`\gl@p` ..... 279, 280,  
287, 288, 687, 713, 731, 740,  
1181, 1182, 1370, 1374, 1389,  
1419, 1431, 2169, 2175, 2182, 2188  
`\goalfraction` ..... 7, 2302
- H**
- `\hangingsymbol` ..... 12, 1562, 1568  
`\hb@xt@` ..... 990, 999, 1009, 1061,  
1071, 1821, 2061, 2064, 2096, 2099  
`\hsize` ..... 863, 906, 1821, 1922–  
1925, 1928, 1942, 1944, 1945,  
1947, 1949, 1960, 1971–1974,  
1977, 1987, 2061, 2064, 2096, 2099
- I**
- `\if@filesw` ..... 1697  
`\if@firstcolumn` 1274, 1299, 1320, 1488  
`\if@ledgroup` ..... 606  
`\if@nobreak` ..... 828, 872  
`\if@noeled@sec` ..... 68, 94  
`\if@nopb` ..... 1885, 1900  
`\if@pb` ..... 1884, 1901  
`\if@pstarts` ... 1739, 1793, 2022, 2038  
`\if@RTL` ..... 682, 693, 708, 719, 1022  
`\ifaraw@text` ... 1751, 1799, 2047, 2119  
`\ifautopar` ..... 855, 899, 1410  
`\ifbypage@` ..... 360  
`\ifbypage@R` ..... 143, 350, 1160  
`\ifbypstart@` ..... 581, 1842, 2330  
`\ifbypstart@R` ... 143, 585, 1846, 2376  
`\ifdefstring` ..... 1014, 1015,  
1804, 1815, 1822, 1832, 2052, 2087  
`\ifdim` ... 624, 628, 630, 634, 1865,  
1871, 2061, 2096, 2145, 2219, 2240  
`\ifdimgreater` ..... 2482, 2485  
`\ifdimless` ..... 2495  
`\iffirst@linenum@out@R` ... 596, 600  
`\ifhbox` ..... 1824, 1829  
`\ifinserthangingsymbol` .....  
..... 1560, 1574, 2283  
`\ifinserthangingsymbolR` .....  
..... 1558, 1566, 1584, 2294  
`\ifinstanzaL` 776, 776, 1561, 1573, 2281  
`\ifinstanzaR` 776, 777, 1567, 1583, 2292  
`\ifl@d@dash` ..... 1399  
`\ifl@d@elin` ..... 1401, 1402  
`\ifl@d@esl` ..... 1402  
`\ifl@d@pnum` ..... 1396, 1400  
`\ifl@d@ssub` ..... 1398  
`\ifl@dpagefull` ..... 2075, 2110, 2211  
`\ifl@dpageing` ..... 13, 1572, 1582  
`\ifl@dpairing` ..... 13, 77, 1410  
`\ifl@dsamelang` ..... 1694  
`\ifl@dsamepage` ..... 2051, 2085, 2211  
`\ifl@dskipnumber` ..... 1251  
`\ifl@dusedbabel` ..... 1692  
`\iflabelpstart` ..... 865, 908  
`\ifledgroupnotesL@` ..... 1100  
`\ifledgroupnotesR@` ..... 1082, 1250  
`\iflednopbinverse` ..... 2282, 2293  
`\ifledplinenum` ..... 1397  
`\ifledRcol` ..... 13, 179,  
201, 205, 209, 213, 257, 274,  
338, 347, 372, 386, 403, 420,  
432, 440, 461, 506, 533, 542,  
551, 625, 631, 637, 644, 652,  
657, 661, 666, 677, 703, 726,  
1414, 1453, 1520, 1533, 1720, 1732  
`\ifluatex` ..... 1534, 1545  
`\ifnoteschanged@` ..... 87  
`\ifnumberedpar@` .....  
... 838, 882, 918, 939, 1518, 1532  
`\ifnumbering` ..... 148, 834, 915  
`\ifnumberingR` ... 43, 74, 112, 878, 936  
`\ifnumberline` ..... 1082, 1100, 1250  
`\ifnumberpstart` ... 856, 900, 927, 948  
`\ifnumequal` ..... 1475  
`\ifnumgreater` ..... 1483, 1503, 1510  
`\ifodd` ..... 1284, 1309,  
1330, 1498, 2143, 2149, 2152, 2159  
`\ifparledgroup` ..... 2471, 2503  
`\ifprint@last@after@pendL` .....  
..... 957, 2067, 2229  
`\ifprint@last@after@pendR` .....  
..... 957, 2102, 2250  
`\ifpst@rtedL` ..... 38, 842  
`\ifpst@rtedR` ..... 38, 886  
`\ifpstartnum` ..... 1340, 1345  
`\ifpstartnumR` ..... 1295  
`\ifshiftedpstarts` 5, 2060, 2095, 2303  
`\ifsidepstartnum` 857, 901, 1297, 1318  
`\ifstrempty` ..... 867, 910, 931, 952

- \IfStrEq ..... 981, 1052, 1886,  
1894, 2258, 2262, 2270, 2274,  
2285, 2286, 2296, 2297, 2421,  
2422, 2429, 2430, 2442, 2443, 2450
  - \ifstrequal ..... 2474, 2480
  - \ifsublines@ ..... 233,  
326, 371, 404, 411, 442, 451,  
463, 470, 482, 516, 572, 574,  
1083, 1101, 1167, 1254, 1455, 1459
  - \ifvbox 974, 1045, 1755, 1758, 2316, 2362
  - \ifvmode ..... 1421, 1433
  - \ifwidthliketwocolumns ..... 11
  - \ifwrittenlinesL ..... 2312, 2324
  - \ifwrittenlinesR ..... 2313, 2370
  - \initnumbering@sectcmd .... 749, 763
  - \initnumbering@sectcountR .....  
..... 66, 99, 119, 2086
  - \InputIfFileExists ..... 69
  - \insert@count 548, 678, 704, 1527, 1554
  - \insert@countR 549, 677, 703, 1523, 1543
  - \insert@noterule@ledgroup .....  
..... 2425, 2433, 2441
  - \inserthangingsymbolfalse ..... 987
  - \inserthangingsymbolL ... 1009, 1558
  - \inserthangingsymbolR ..... 1558
  - \inserthangingsymbolRfalse .... 1058
  - \inserthangingsymbolRtrue .... 1056
  - \inserthangingsymboltrue ..... 985
  - \insertlines@listR .....  
.... 80, 243, 259, 554, 1366, 1370
  - \inserts@list ..... 844, 1526, 1553
  - \inserts@listR ..... 888, 1361,  
1364, 1374, 1388, 1389, 1522, 1542
  - \istanzaLfalse ..... 1861, 2134
  - \istanzaLtrue ..... 787
  - \istanzaRfalse ..... 1862, 2135
  - \istanzaRtrue ..... 810
  - \interlinepenalty ..... 1601
  - \itemcount@ ..... 1473, 1475,  
1480, 1483, 1501, 1503, 1508, 1510
- L**
- \l@d@nums ..... 729, 732, 738, 741
  - \l@d@set ..... 419, 652, 653
  - \l@d@bbl@set@language .... 1694, 1717
  - \l@d@bfnote ..... 1517
  - \l@d@maxchunks ..... 850, 852,  
894, 896, 1651, 1661, 1669, 1676
  - \l@d@calc@maxoftwo .....  
..... 2027, 2198, 2345, 2387
  - \l@d@calc@minoftwo .. 2035, 2122, 2198
  - \l@d@calcnum ..... 1188
  - \l@d@checklang ..... 1694
  - \l@d@chset@num ..... 293, 296, 419
  - \l@d@csnotetext ..... 1482, 1485
  - \l@d@csnotetext@l ..... 1485, 1502
  - \l@d@csnotetext@r ..... 1485, 1509
  - \l@d@emptyd@ta ..... 978, 1049
  - \l@d@end@stuff ..... 63, 117, 133, 141
  - \l@d@getline@margin ..... 177
  - \l@d@getsidenote@margin ..... 1464
  - \l@d@d@ta ..... 1006,  
1275, 1287, 1300, 1312, 1321, 1333
  - \l@d@leftbox .....  
.. 959, 989, 999, 1823, 2061, 2064
  - \l@d@linenumR ..... 225
  - \l@d@lsn@te ..... 1008
  - \l@d@make@labels ..... 1437
  - \l@d@make@labelsR ..... 1425, 1442
  - \l@d@minpagelines .....  
.... 1996, 2036, 2123, 2220, 2241
  - \l@d@numpstartsL ..... 849,  
850, 852, 854, 868, 869, 932,  
933, 1655, 1687, 1742, 1781,  
1782, 1858, 2007, 2008, 2132, 2328
  - \l@d@numpstartsR ..... 47, 893,  
894, 896, 898, 911, 912, 953,  
954, 1655, 1688, 1745, 1781,  
1782, 1859, 2007, 2008, 2133, 2374
  - \l@d@d@bbl@set@language ..... 1716
  - \l@d@d@selectlanguage 1715, 1719, 1724
  - \l@d@pagefullfalse .....  
.... 2211, 2260, 2265, 2272, 2277
  - \l@d@pagefulltrue .....  
.... 2211, 2259, 2264, 2271, 2276
  - \l@d@pagingfalse ..... 14, 748, 768
  - \l@d@pagingtrue ..... 762
  - \l@d@pairingfalse ..... 13, 752, 767
  - \l@d@pairingtrue ..... 747, 761
  - \l@d@pscL ..... 974, 979,  
992, 1016, 1019, 1021, 1657,  
1689, 1742, 1755, 1790, 1796,  
1802, 1805, 1807, 1856, 2017,  
2023, 2028, 2031, 2039, 2053,  
2054, 2056, 2068, 2069, 2130,  
2316, 2318, 2321, 2328, 2339,  
2340, 2345, 2347, 2349, 2420, 2475
  - \l@d@pscR ..... 1045, 1050, 1063,  
1658, 1690, 1745, 1758, 1791,  
1797, 1813, 1816, 1818, 1857,

- 2018, 2024, 2032, 2040, 2088,  
2089, 2091, 2103, 2104, 2131,  
2362, 2364, 2367, 2374, 2383,  
2384, 2387, 2389, 2391, 2428, 2477  
\l@drd@ta ..... 1009,  
1277, 1285, 1302, 1310, 1323, 1331  
\l@drightbox .....  
959, 1060, 1071, 1828, 2096, 2099  
\l@drsn@te ..... 1010  
\l@dsamepagefalse .....  
.... 2211, 2259, 2264, 2271, 2276  
\l@dsamepagetrue .....  
.... 2211, 2260, 2265, 2272, 2277  
\l@dsetupmaxlinecounts .. 1668, 1685  
\l@dsetupprawboxes ..... 1660, 1684  
\l@dskipnumberfalse ..... 1252  
\l@dskipnumbertrue ..... 1148  
\l@dunhbox@line .... 1009, 1024, 1027  
\l@dusedbabelfalse ..... 1692, 1712  
\l@dusedbabeltrue ..... 1692, 1714  
\l@duselanguage .....  
.... 1704, 1800, 1811, 2049, 2084  
\l@dzeromaxlinecounts ... 1668, 1686  
\l@dzeropenalties 921, 942, 1785, 2012  
\l@prev@nopbR ... 54, 2407, 2414, 2415  
\l@prev@pbR ..... 53, 2406, 2412, 2413  
\l@pscl ..... 1657  
\l@pscr ..... 1657  
\label@refs .....  
1417, 1419, 1425, 1429, 1431, 1437  
\labelref@list ..... 1428, 1431, 1460  
\labelref@listR 1407, 1416, 1419, 1456  
\language name .. 1695, 1696, 1698–1701  
\last@page@num ..... 358, 364  
\last@page@numR ..... 344  
\lastbox ..... 982, 1053  
\lastskip ..... 623, 629  
\Lcolwidth .....  
.. 6, 7, 16, 764, 863, 990, 999,  
1808, 1923, 1944, 1961, 1972, 1988  
\led@err@BadLeftRightPstarts ...  
..... 29, 1782, 2008  
\led@err@LeftOnRightPage ... 32, 2153  
\led@err@LineationInNumbered ... 149  
\led@err@ManyLeftnotes ..... 1503  
\led@err@ManyRightnotes ..... 1510  
\led@err@ManySidenotes ..... 1483  
\led@err@NumberingNotStarted ... 91  
\led@err@numberingShouldHaveStarted  
..... 121  
\led@err@NumberingStarted ..... 44  
\led@err@PendNoPstart ..... 919, 940  
\led@err@PendNotNumbered ... 916, 937  
\led@err@PstartInPstart ... 839, 883  
\led@err@PstartNotNumbered . 835, 879  
\led@err@RightOnLeftPage ... 32, 2160  
\led@err@TooManyPstarts .....  
..... 24, 26, 851, 895  
\led@mess@NotesChanged ..... 88  
\led@mess@SectionContinued .....  
..... 115, 131, 139  
\led@nopbnumR ..... 2411, 2412  
\led@nopbR ..... 2410, 2412  
\led@pb@setting .....  
..... 1886, 1894, 2258, 2262,  
2270, 2274, 2285, 2286, 2296, 2297  
\led@pbnumR ..... 2409, 2412  
\led@pbR ..... 2408, 2412  
\led@warn@BadAction ..... 1150  
\led@warn@BadAdvancelineLine 389, 395  
\led@warn@BadAdvancelineSubline .  
..... 375, 381  
\led@warn@BadLineation ..... 166  
\led@warn@BadSetline ..... 642  
\led@warn@BadSetlinenum ..... 650  
\led@warn@DuplicateLabel ..... 1444  
\ledgroupnotesL@false ..... 2436  
\ledgroupnotesL@true ..... 2424  
\ledgroupnotesR@false ..... 2439  
\ledgroupnotesR@true ..... 2432  
\ledllfill ..... 1009  
\lednopb ..... 806  
\lednopbnum ..... 2285, 2408  
\lednopbnumR ..... 2296, 2408  
\lednopbR ..... 806, 2410  
\ledpb ..... 805  
\ledpbnum ..... 2286  
\ledpbnumR ..... 2297, 2408  
\ledpbR ..... 805, 2408  
\ledRcol@false ..... 1074  
\ledRcol@true ..... 1043  
\ledRcolfalse ..... 15, 779, 812  
\ledRcoltrue ..... 796  
\ledrlfill ..... 1009, 1932, 1954  
\ledsavedprintlines ..... 10, 1394  
\ledsectnomark ..... 1015  
\ledsectnotoc ..... 1014, 2052, 2087  
\ledstrutL ..... 2061, 2064, 2138  
\ledstrutR ..... 2096, 2099, 2138  
\ledthegoal ..... 2219, 2240, 2302

- \leftlinenumR ..... 225, 1275, 1287
  - \leftstartnumL ..... 1295
  - \leftstartnumR ..... 1295
  - Leftside (environment) ..... 7, 778
  - \Leftsidehook ..... 785, 790
  - \Leftsidehookend ..... 789, 790
  - \line@list ..... 736, 740
  - \line@list@stuff ..... 132
  - \line@list@stuffR .. 62, 116, 140, 598
  - \line@listR . 83, 243, 258, 574, 727, 731
  - \line@margin ..... 182, 1305
  - \line@marginR ..... 175, 1280, 1326
  - \line@num . 361, 393, 394, 396, 414, 425, 426, 454, 581, 1107, 1458, 2333
  - \line@numR ..... 55, 232, 239, 298, 332, 351, 387, 388, 390, 407, 421, 422, 445, 567, 571, 585, 1089, 1161, 1170, 1259, 1261, 1263, 1264, 1454, 2379
  - \lineation ..... 171, 172, 807
  - \lineation\* ..... 8, 171
  - \lineationR ..... 8, 147, 173, 807
  - \linenum@out ..... 620, 626, 632, 638, 645, 653, 658, 662, 1427, 1766, 1843, 2192
  - \linenum@outR ..... 595, 601, 603, 607, 608, 610, 611, 614, 615, 622, 625, 631, 637, 644, 652, 657, 661, 666, 1415, 1771, 1847, 2195, 2408–2411
  - \linenumberlist ..... 1260, 1264
  - \linenumincrement ..... 7, 196
  - \linenumincrement\* ..... 7, 196
  - \linenummargin ..... 175
  - \linenumr@p .... 1397, 1401, 1454, 1458
  - \linenumrepR ..... 222, 232
  - \linenumsep ..... 227, 229, 1342, 1345, 1354, 1357
  - \linesinpar@listL ..... 250, 268, 582, 2166, 2169
  - \linesinpar@listR ..... 250, 262, 586, 2172, 2175
  - \linesonpage@listL 269, 590, 2179, 2182
  - \linesonpage@listR 263, 593, 2185, 2188
  - \list@clear ..... 258–265, 268, 269, 271, 844, 888
  - \list@clearing@reg ..... 267
  - \list@create ..... 243–248, 250–252, 1361, 1407
  - \listxadd ..... 366, 2412–2415
  - \lock@disp ..... 1220, 1224, 1229
  - \lock@off .... 503, 504, 512, 661, 662
  - \lock@on ..... 657, 658
- M**
- \managestanza@modulo ..... 1627
  - \maxchunks ..... 5, 1651
  - \maxlinesinpar@list ..... 250, 271
  - \memorydump ..... 9, 784, 801
  - \memorydumpL ..... 126, 784
  - \memorydumpR ..... 126, 801
  - \message ..... 61
  - \multiply ..... 1194, 1926, 1975
- N**
- \n@num ..... 540, 666
  - \n@num@reg ..... 546
  - \namebox ..... 974, 979, 1045, 1050, 1635, 1755, 1758, 2316, 2362
  - \NeedsTeXFormat ..... 2
  - \new@line ..... 1024, 1027
  - \new@lineL ..... 619, 1009
  - \new@lineR ..... 621
  - \newbox ..... 817, 959, 960, 1636
  - \newcommandx ..... 827, 871, 914, 935
  - \newcounter ..... 99–102, 187, 189, 191, 193, 820, 821, 823, 824
  - \newif ..... 5, 39, 143, 144, 596, 776, 777, 957, 958, 1349, 1558, 1692, 1739, 1751, 1884, 1885, 2211, 2213, 2312, 2313
  - \newlength ..... 1912, 1915
  - \newmarks ..... 2416–2418
  - \newnamebox ..... 1635, 1663, 1664
  - \newnamecount ..... 1646, 1671
  - \newsavebox ..... 1775, 1776
  - \newwrite ..... 595, 2405
  - \next@absline ..... 1887, 1889, 1891, 2263–2265
  - \next@abslineR ..... 1888, 1890, 1892, 2275–2277
  - \next@action ..... 288
  - \next@actionline ..... 285, 287
  - \next@actionlineR ..... 277, 279, 1119, 1157, 1179, 1181
  - \next@actionR ..... 280, 1120, 1158, 1159, 1164, 1165, 1173, 1182
  - \next@insert ..... 845
  - \next@insertR ..... 889, 1365, 1368, 1370, 1373, 1377

- \next@page@num ..... 365, 437
  - \next@page@numR 59, 301, 303, 355, 434
  - \no@expands ..... 675, 701
  - \noindent ..... 869, 912, 933, 954
  - \normal@page@break ..... 366
  - \normal@page@breakR ..... 52
  - \normal@pars ..... 76, 848, 892
  - \normalbfnoteX ..... 1531
  - \notefontsetup ..... 2462
  - \noteschanged@true .....
    - ..... 81, 84, 728, 737, 1367
  - \notesXwidthliketwocolumns ..... 6
  - \num@lines ..... 922, 1786, 2013
  - \num@linesR ..... 816, 943, 1787, 2014
  - \numberedpar@true ..... 864, 907
  - \numberingRfalse ..... 75
  - \numberingRtrue ..... 49, 110, 136
  - \numberingtrue ..... 128
  - \numberpstartfalse ..... 10
  - \numberpstarttrue ..... 10
  - \numdef ..... 1016, 1501, 1508,
    - 1887, 1888, 2263, 2275, 2475, 2477
  - \numgdef ..... 1473, 1480, 2284, 2295
  - \numlabfont ..... 232
  - \numpagelinesL ..... 1996,
    - 2059, 2076, 2080, 2220, 2286, 2495
  - \numpagelinesR .....
    - 1996, 2094, 2111, 2115, 2241, 2297
- O**
- \old@otherlanguage ..... 1729
  - \old@startstanza .. 786, 787, 809, 810
  - \oldchapter ..... 759, 769
  - \one@line .. 979, 982, 1009, 1024, 1027
  - \one@lineR ..... 816, 1050, 1053
  - \openout ..... 70, 603, 608, 611, 615
  - \otherlanguage ..... 1729, 1730
- P**
- \p@pstartL ..... 866
  - \p@pstartR ..... 909
  - \PackageError ..... 21
  - \page@action ..... 302, 431, 560
  - \page@num ..... 283, 363, 1307
  - \page@numR ..... 254, 275, 353,
    - 566, 571, 1159, 1282, 1328, 1496
  - \pagebreak ..... 1901
  - \pagegoal ..... 2310
  - \Pages ..... 7, 2000
  - pages (environment) ..... 6, 746
  - \pagetotal ..... 2145, 2219, 2240
  - pairs (environment) ..... 5, 746
  - \paperwidth ..... 1023, 1026
  - \par@line ..... 923, 1788, 2015
  - \par@lineR ..... 816, 944, 1789, 2016
  - \parbox ..... 1808, 1819
  - \parledgroup@ .. 981, 1052, 2416, 2442
  - \parledgroup@beforenotes@save ..
    - ..... 930, 951, 2502
  - \parledgroup@beforenotesL ..... 2500
  - \parledgroup@beforenotesR ..... 2500
  - \parledgroup@correction@notespacing
    - ..... 2063, 2098, 2489
  - \parledgroup@correction@notespacing@final
    - ..... 2351, 2393, 2470
  - \parledgroup@correction@notespacing@init
    - ..... 2081, 2116, 2465, 2473
  - \parledgroup@notes@endL .....
    - ..... 2319, 2322, 2350, 2435
  - \parledgroup@notes@endR .....
    - ..... 2365, 2368, 2392, 2438
  - \parledgroup@notes@startL .....
    - ..... 981, 2419, 2435
  - \parledgroup@notes@startR .....
    - ..... 1052, 2419, 2435
  - \parledgroup@notespacing@correction
    - ..... 2460, 2492–2494
  - \parledgroup@notespacing@correction@accumulated
    - ..... 2466, 2472, 2493
  - \parledgroup@notespacing@correction@modulo
    - ..... 2467, 2494–2496
  - \parledgroup@notespacing@set@correction
    - ..... 2004, 2460
  - \parledgroup@series .....
    - ..... 2417, 2421, 2422,
      - 2429, 2430, 2445, 2446, 2452, 2453
  - \parledgroup@type ..... 2418,
    - 2421, 2422, 2429, 2430, 2443, 2450
  - \parledgroupnotespacing .. 2459, 2462
  - \parledgroupseries@ ..... 2416
  - \parledgrouptrue ..... 10
  - \parledgrouptype@ ..... 2416
  - \pausenumbering ..... 799
  - \pausenumberingR ..... 109, 799
  - \pend ..... 8, 783, 804, 840, 1614
  - \pendL ..... 783, 914
  - \pendR ..... 804, 884, 935
  - \prev@abslineverse .....
    - ..... 2284–2286, 2295–2297
  - \prev@nopbR ..... 2406

`\prev@pbR` ..... 2406  
`\prevgraf` .....  
     . 922, 943, 1786, 1787, 2013, 2014  
`\print@columnseparator` .. 1827, 1864  
`\print@eledsectionL` .....  
     ..... 1004, 1012, 1808, 2057  
`\print@eledsectionR` . 1078, 1819, 2092  
`\print@last@after@pendLfalse` .. 2232  
`\print@last@after@pendLtrue` ... 2355  
`\print@last@after@pendRfalse` .. 2253  
`\print@last@after@pendRtrue` ... 2395  
`\print@lineL` ..... 994, 1004  
`\print@lineR` ..... 1065, 1078  
`\printlines` ..... 1405  
`\printlinesR` ..... 10, 1394  
`\ProcessOptions` ..... 12  
`\protected@edef` ..... 865, 908  
`\protected@write` ... 1424, 1436, 1698  
`\protected@xdef` ..... 1540, 1551  
`\ProvidesPackage` ..... 3  
`\pst@rtedLfalse` ..... 38  
`\pst@rtedLtrue` ..... 129, 846  
`\pst@rtedRfalse` ..... 40, 48, 78  
`\pst@rtedRtrue` ..... 113, 137, 890  
`\pstart` ..... 8, 27, 31, 781, 803, 1616  
`\pstartL` ..... 781, 819  
`\pstartnumfalse` ..... 1342, 1347  
`\pstartnumRfalse` ..... 1354, 1359  
`\pstartnumRtrue` .... 1350, 1795, 2386  
`\pstartnumtrue` ..... 1794, 2344  
`\pstartR` ..... 803, 819

## R

`\Rcolwidth` .....  
     6, 7, 16, 765, 906, 1061, 1071,  
     1819, 1924, 1945, 1962, 1973, 1989  
`\read@linelist` ..... 256, 599  
`\rem@inder` ..... 1264, 1266–1268  
`\resetprevline@` 1844, 1848, 2334, 2380  
`\resumenumbering` ..... 800  
`\resumenumberingR` ..... 109, 800  
`\rightlinenumR` ..... 225, 1277, 1285  
`\rightpstartnumL` ..... 1295  
`\rightpstartnumR` ..... 1295  
`Rightside (environment)` ..... 7, 795  
`\Rightsidehook` ..... 790, 808  
`\Rightsidehookend` ..... 790, 813  
`\rlap` 1277, 1285, 1302, 1310, 1323, 1331  
`\Rlineflag` 9, 220, 232, 1397, 1401, 1446  
`\rule` ..... 1904

## S

`\savebox` ..... 1808, 1819  
`\sc@n@list` ..... 1265, 1267  
`\secdef` ..... 774  
`\section@num` ..... 130–132  
`\section@numR` ..... 36,  
     50, 61, 62, 69, 70, 114–116, 138–140  
`\select@language` 1696, 1698–1701, 1735  
`\selectlanguage` ..... 1704  
`\set@line` ..... 676, 702, 725  
`\set@line@action` .....  
     ..... 295, 400, 409, 416, 439, 562  
`\setl@dlp@rbox` ..... 1489, 1504, 1506  
`\setl@drp@rbox` ..... 1491, 1499, 1511  
`\setline` ..... 640  
`\setlinenum` ..... 648  
`\setnamebox` ..... 854, 898, 1635  
`\setnotepositionliketwocolumns@C`  
     ..... 1918  
`\setnotepositionliketwocolumns@L`  
     ..... 1918  
`\setnotepositionliketwocolumns@R`  
     ..... 1918  
`\setnotespositionliketwocolumns@C`  
     ..... 1957  
`\setnotespositionliketwocolumns@L`  
     ..... 1935  
`\setnotespositionliketwocolumns@R`  
     ..... 1984  
`\setpositionliketwocolumns@C` ...  
     ..... 1918, 1952  
`\setpositionliketwocolumns@L` ...  
     ..... 1918, 1931  
`\setpositionliketwocolumns@R` ...  
     ..... 1918, 1980  
`\setprintlines` ..... 1395  
`\setwidthliketwocolumns@C` 1918, 1939  
`\setwidthliketwocolumns@L` 1918, 1918  
`\setwidthliketwocolumns@R` 1918, 1969  
`\shiftedpstartsfalse` ..... 7  
`\shiftedpstartstrue` ..... 6, 8, 9  
`\shiftedversesfalse` ..... 7  
`\shiftedversestrue` ..... 6  
`\showlemma` ..... 685, 711  
`\sidenote@margin` ..... 1466, 1469  
`\sidenote@marginR` ..... 1463, 1494  
`\sidenotecontent@` .....  
     . 1472, 1477, 1478, 1489, 1491,  
     1499, 1500, 1504, 1506, 1507, 1511  
`\sidenotemargin` ..... 1463



- `\sidenotemargin*` ..... 1463  
`\sidenotesepp` ..... 1478  
`\skip` ..... 2445, 2452  
`\skip@lockoff` ..... 504, 512  
`\skipnumbering` ..... 10, 665  
`\skipnumbering@reg` ..... 669  
`\smash` ..... 1904  
`\splitbotmarks` ..... 2442,  
2443, 2445, 2446, 2450, 2452, 2453  
`\splitfirstmarks` .....  
981, 1052, 2421, 2422, 2429, 2430  
`\splittopskip` ..... 976, 1047  
`\stanza@count` ..... 1596, 1610, 1622  
`\stanza@hang` ..... 1598, 1630  
`\stanza@modulo` ..... 1596, 1625  
`\stanzaindentbase` .....  
..... 1576, 1586, 1623, 1626  
`\startlock` ..... 656  
`\startstanzahook` ..... 1594  
`\startsub` ..... 623  
`\sub@action` ..... 311, 460, 561  
`\sub@change` ..... 60, 305, 306, 312  
`\sub@lock` ..... 1102  
`\sub@lockR` ..... 57, 320, 322, 324,  
327, 478, 484, 485, 487, 488,  
518, 519, 521, 1084, 1140, 1142,  
1143, 1145, 1201, 1241, 1243, 1245  
`\sub@off` ..... 631, 632  
`\sub@on` ..... 625, 626  
`\subline@num` ..... 234, 361, 379,  
380, 382, 412, 452, 1103, 1108, 1459  
`\subline@numR` ..... 235,  
239, 328, 332, 351, 373, 374,  
376, 405, 443, 568, 572, 1085,  
1090, 1161, 1168, 1255, 1256, 1455  
`\sublinenumincrement` ..... 7, 196  
`\sublinenumincrement*` ..... 7, 196  
`\sublinenumr@p` . 1398, 1402, 1455, 1459  
`\sublinenumrepR` ..... 222, 235  
`\sublines@false` ..... 58, 309, 1130  
`\sublines@true` ..... 307, 1128  
`\sublock@disp` ..... 1203, 1207, 1212  
`\sw@list@inedtextR` ..... 248, 265  
`\sw@listR` ..... 247, 264  
`\symplinum` ..... 1397  
`\sza@penalty` ..... 1605, 1609
- T**
- `\temp` ..... 1921, 1922, 1925–1928,  
1941, 1942, 1947–1949, 1958,  
1960, 1963–1966, 1970, 1971,  
1974–1977, 1985, 1987, 1990–1993  
`\temp@` ..... 1016, 1017  
`\temp@spacing` ..... 2462, 2463  
`\tempa` 1959, 1961–1963, 1986, 1988–1990  
`\textwidth` ..... 17, 19, 764, 765  
`\theledlanguageL` ... 1704, 1800, 2049  
`\theledlanguageR` ... 1704, 1811, 2084  
`\thepage` ..... 620, 622, 1425, 1437  
`\thepstart` ..... 782, 802  
`\thepstartL` .....  
10, 782, 822, 859, 866, 1341, 1346  
`\thepstartR` .....  
10, 802, 825, 902, 909, 1353, 1358  
`\thisfootnote` .. 1540, 1541, 1551, 1552  
`\thr@@` .. 487, 496, 519, 526, 1135, 1143  
`\topskip` ..... 2145
- U**
- `\unhbox` ..... 1640,  
1823, 1828, 2061, 2064, 2096, 2099  
`\unhnamebox` ..... 1635  
`\unvbox` ..... 982, 1053, 1642  
`\unvnamebox` ..... 1635  
`\usebox` ..... 1825, 1830  
`\usernamecount` 1597, 1604, 1646, 1678,  
2028, 2318, 2321, 2345, 2347,  
2364, 2367, 2387, 2389, 2420, 2428
- V**
- `\value` ..... 843, 887, 1621,  
1779, 1780, 2002, 2003, 2331, 2377  
`\vbadness` ..... 975, 1046  
`\vbfnoteX` ..... 1541, 1552  
`\vbox` ..... 854, 898, 1808, 1819  
`\vl@dbfnote` ..... 1521, 1525  
`\vsplit` ..... 979, 1050
- W**
- `\wd` ..... 1009  
`\widthliketwocolumns` ..... 6  
`\widthliketwocolumnstrue` ..... 11  
`\WithSuffix` ..... 171, 216–219, 1463  
`\writtenlinesLfalse` ..... 2019, 2329  
`\writtenlinesLtrue` ..... 2326  
`\writtenlinesRfalse` ..... 2020, 2375  
`\writtenlinesRtrue` ..... 2372
- X**
- `\x@lemma` ..... 687–689, 713–715

<code>\xifinlist</code> .....	992,	.....	433, 434, 436, 437,
	1017, 1063, 1802, 1813, 2056, 2091		441, 448, 450, 457, 462, 464,
<code>\xifinlistcs</code> .....	1889–		466, 469, 471, 473, 481, 483,
	1892, 1895–1898, 2259, 2260,		492, 515, 517, 524, 543, 544,
	2264, 2265, 2271, 2272, 2276, 2277		554, 570, 582, 586, 590, 593,
<code>\Xnoteswidthliketwocolumns</code> .....	6		1454, 1458, 1521, 1525, 1541, 1552
<code>\xpg@main@language</code> .....	1736, 1737		
<code>\xright@appenditem</code> .....		<b>Z</b>	
		<code>\zz@@@</code> .....	1417, 1429

## Change History

v0.1		Pstart number can be printed in side .....	80
General: First public release .....	1		
v0.10		v0.12	
General: <code>\edlabel</code> commands on the right side are now correctly indicated. ....	1	General: New new management of hangingsymbol insertion, preventing undesirable insertions. ....	58
<code>\edlabel</code> commands which start a paragraph are now put in the right place. ....	1	v0.2	
v0.11		General: Added section of babel related code .....	62
General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by pstart (like in eledmac 0.15). ....	43	Fix babel problems .....	1
Lineation can be by pstart (like in eledmac 0.15). ....	19	<code>\Columns</code> : Added <code>\l@dcchecklang</code> and <code>\l@duselanguage</code> to <code>\Columns</code> .....	66
New management of hangingsymbol insertion, preventing undesirable insertions. ....	58	<code>\Pages</code> : Added <code>\l@duselanguage</code> to <code>\Pages</code> .....	73
Prevent shift of column separator when a verse is hanged .....	58	v0.3	
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in eledmac 0.15). ....	48	General: Added <code>\do@lineLhook</code> and <code>\do@lineRhook</code> .....	44
<code>\Columns</code> : Line numbering by pstart. ....	67	Reorganize for ledarab .....	1
<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in eledmac 0.15). ....	80	<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to match new eledmac .....	48
		<code>\do@actions@nextR</code> : Used <code>\do@actions@fixedcode</code> in <code>\do@actionsR</code> .....	47
		<code>\do@lineL</code> : Added <code>\do@lineLhook</code> to <code>\do@lineL</code> .....	43
		Simplified <code>\do@lineL</code> by using macros for some common code .....	43
		<code>\do@lineR</code> : Changed <code>\do@lineR</code> similarly to <code>\do@lineL</code> .....	45

Leftside: Added hooks into Left- side environment . . . . .	37	v0.4	General: No more ledparpatch. All patches are now in the main file. . . . .	1
\flag@end: Removed extraneous spaces from \flag@end . . . . .	33	v0.5	General: Corrections about \section and other titles in numbered sections . . . . .	1
\ifledRcol: Moved \ifl@dpairing toeledmac . . . . .	15	v0.6	General: Be able to use \chapter in parallel pages. . . . .	1
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac . . . . .	16	v0.7	General: Option ‘shiftedverses’ which make there is no blank between two parallel verses with inequal length. . . . .	1
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR . . . . .	22	v0.8	General: Possibility to have a sym- bol on each hanging of verses, like in the french typogra- phy. Redefine the commande \hangingsymbol to define the character. . . . .	1
\l@dnumpstartsR: Moved \l@dnumpstartsL toeledmac . . . . .	61	v0.9	General: Possibility to number \pstart. . . . .	10
\ledsavedprintlines: Simpli- fied \printlinesR by using \setprintlines . . . . .	54		Possibility to number the pstart with the commands \numberpstarttrue. . . . .	1
\ledstrutR: Added \ledstrutL and \ledstrutR . . . . .	75		\ifledRcol: Moved \iflledRcol and \ifnumberingR toeledmac . . . . .	15
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX . . . . .	57	v0.9.1	General: The numbering of the pstarts restarts on each \beginnumbering. . . . .	1
\Pages: Added \ledstrutL to \Pages . . . . .	73	v0.9.2	General: Debug : with \Columns, the hanging indentation now runs on the left columns and the hanging symbol is shown only when \stanza is used. . . . .	1
Added \ledstrutR to \Pages . . . . .	74	v0.9.3	General: \thepstartL and \thepstartR use now \bfseries and not \bf, which is deprecated and makes con- flicts with memoir class. . . . .	1
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend . . . . .	38	v1.0	General: Compatibility witheled- mac. Change name toeledpar. . . . .	1
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR . . . . .	21			
v0.3.a				
General: Minor \linenummargin fix . . . . .	1			
v0.3.b				
General: Improved parallel page balancing . . . . .	1			
v0.3.c				
General: Compatibilty with Poly- glossia . . . . .	1			
v0.3a				
\line@marginR: Don’t just set \line@marginR in \linenummargin . . . . .	20			
v0.3b				
\Pages: Added \l@dminpagelines calculation for succeeding page pairs . . . . .	74			

Debug in lineation by pstart . . .	19	<code>\l@dbfnote</code> : Spurious space with footnote in right column. . . .	57
v1.0.1			
General: Correction on <code>\numberonlyfirstline</code> with lineation by pstart or by page. . . . .	1	General: Allow use of commands in sidenotes, as introduced by eledmac 1.0. . . . .	56
v1.1			
General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code> . . . . .	1	v1.3.5	
<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from eledmac). . . . .	39	<code>\normalbfnoteX</code> : Allows one to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded. . . . .	57
v1.1.1			
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1. . . . .	39	v1.4	
v1.1.2		General: Added <code>\do@insidelineLhook</code> and <code>\do@insidelineRhook</code> . . .	44
<code>\affixside@noteR</code> : Remove spurious space between line number and line content . . . . .	56	v1.4.1	
v1.10.0		<code>\normalbfnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text. . .	57
General: Compatibility with <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> . . . . .	1	<code>astanza</code> : Enable the use of <code>stanzaindent</code> repetition within <code>astanza</code> environment. . . . .	59
Restore critical notes in <code>\eledsection</code> in parallel columns (this bug was added in 1.8.2). . . . .	1	v1.4.2	
<code>\edlabel</code> : Prevent some bugs with cross-referencing when using <code>\autopar</code> in parallel texts. . .	54	<code>\line@list@stuffR</code> : Open / close immediately the line-list file when in minipage, except if the minipage is a ledgroup. . . . .	32
<code>\Pages</code> : Debug wrong pages splitting when no optional argument is used in last <code>\pend</code> (bug was added in v.1.8.3). . . . .	71	v1.4.3	
Debug wrong parallel pages synchronization when an <code>\edtext</code> falls across two pages. . . . .	71	General: Corrects a false hanging verse when a verse is exactly the length of a line. . . . .	1
v1.2		<code>\inserthangingsymbolR</code> : Hang verse is now not automatically flush right. . . . .	58
General: Support for <code>\led&lt;section&gt;</code> commands in parallel texts. . . .	1	<code>\pendL</code> : Spurious spaces in <code>\pendL</code> . . . . .	41
v1.2.1		<code>\pendR</code> : Spurious spaces in <code>\pstartR</code> . . . . .	42
<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once. . . . .	18	<code>\pstartR</code> : Spurious spaces in <code>\pstartL</code> and <code>\pstartR</code> . . . .	39
v1.3		v1.5.0	
General: Manage RTL language. . .	34	General: Add, as in eledmac, features to manage page breaks. . .	1
v1.3.2		<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. . . . .	21
General: Debug with some classes. .	1	v1.6.0	
v1.3.3		General: Add tool and documentation for parallel ledgroups . . .	12
General: Debugging the left notes of the right column. . . . .	56		

v1.7.0	
General: Add, as in eledmac, features to make crossrefs with pstart numbers. . . . .	1
v1.8.0	
General: <code>\beginnumbering</code> is defined only on eledmac, not on eledpar. . . . .	16
<code>\l@dlsnote</code> , <code>\l@drsnote</code> and <code>\l@dcsnote</code> defined only one time, in eledmac. . . . .	56
Add <code>\beforecolumnseparator</code> and <code>\aftercolumnseparator</code> . . .	6
Add <code>\columnspan</code> . . . . .	6
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@lineL</code> to allow new sectioning commands. . . . .	43
Compatibility with musixtex. . .	1
Debug eledmac sectioning command after using <code>\resumenumbering</code> . . . . .	1
New sectioning commands, as in eledmac. . . . .	14
<code>\Columns</code> : Modify <code>\Columns</code> to enable to add section's title. . .	65
Suppress <code>\l@dchecklang</code> from <code>\Columns</code> . . . . .	66
<code>\l@dchecklang</code> : Suppress <code>\l@dchecklang</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. . . . .	62
<code>\Pages</code> : Modify <code>\Pages</code> to enable to add section's title. . . . .	71
<code>\pendL</code> : As in eledmac, <code>\pendL</code> can have an optional argument. . .	41
<code>\pendR</code> : As in eledmac, <code>\pendR</code> can have an optional argument. . .	42
<code>\print@columnseparator</code> : Move some code of <code>\Columns</code> to <code>\print@columnseparator</code> . . .	67
<code>\pstartR</code> : As in eledmac, <code>\pendL</code> and <code>\pendR</code> can have an optional argument. . . . .	39
<code>\sidenotemargin*</code> : <code>\sidenotemargin</code> is now directly defined in eledmac to be able to manage eledpar. . . . .	56
Add <code>\sidenotemargin*</code> . . . .	56
<code>\theledlanguageR</code> : Correct left/right language setting with polyglossia. . . . .	64
v1.8.1	
<code>\do@lineL</code> : Fix a bug with critical notes a the begining of a page, (maybe added by v1.8.0) (?). .	43
<code>\do@lineR</code> : Fix a bug with critical notes a the begining of a page, added by v1.8.0 (?). . . . .	45
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 eledpar send to eledpar handbook . . . . .	16
<code>\flag@end</code> : <code>\flag@start</code> and <code>\flag@end</code> are now defined only one time for eledmac and eledpar . . . . .	33
<code>\lineation*</code> : Add <code>\lineation*</code> .	20
v1.8.3	
General: Add <code>\noeledxxx</code> , as in eledmac . . . . .	1
<code>\doinsidelineRhook</code> : Added <code>\dolineLhook</code> , <code>\dolineRhook</code> , <code>\doinsidelineLhook</code> and <code>\doinsidelineRhook</code> . . . . .	44
<code>\Pages</code> : Debug blank pages when using optional argument in the last <code>\pend</code> . . . . .	71
<code>\resumenumberingR</code> : Debug <code>\resumenumberingR</code> . . . . .	18
v1.9.0	
General: Add <code>\AtBeginPairs</code> macro. . . . .	5
Compatibility with <code>\Xnoteswidthliketwocolumns</code> and <code>\notesXwidthliketwocolumns</code> . . . . .	1

<code>\ifwidthliketwocolumns:</code>	Added		.aux file when setting left/right	
	widthliketwocolumns option . .	15	lines. . . . .	64
<code>\theledlanguageR:</code>	Debug	v1.9.1		
	left/right language switching		<code>\ifledRcol:</code> Moved <code>\ifl@dpaging</code>	
	with polyglossia. Don't write in		to eledmac . . . . .	15