

# Parallel typesetting for critical editions: the `reledpar` package\*

Maïeul Rouquette<sup>†</sup> based on the original `ledpar` by Peter Wilson  
Herries Press<sup>‡</sup>

## Abstract

The `reledmac` package has been used for some time for typesetting critical editions. The `reledpar` package is an extension to `reledmac` which enables texts and their critical apparatus to be typeset in parallel, either in two columns or on pairs of facing pages.

`reledpar` provides many tools and options. Normally, they are all documented in this file. Also provided is a help folder, “examples”. The folder contains additional examples (although not for all cases). Examples starting by “3-” are for basic uses, those starting by “4-” are for advanced uses.

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

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

## Contents

<b>1 Introduction</b>	<b>5</b>
1.1 Aim of this package . . . . .	5
1.2 Historical overview . . . . .	5
<b>2 General</b>	<b>5</b>
<b>3 Parallel columns</b>	<b>6</b>
3.1 Basic use . . . . .	6
3.2 Setting . . . . .	7
3.2.1 Column’s width . . . . .	7
3.2.2 Column’s separator . . . . .	7
3.2.3 Column’s positions . . . . .	7
3.2.4 Mixing two columns and one column texts . . . . .	8

---

\*This file (`reledpar.dtx`) has version number v2.3.0, last revised 2015/09/05.

<sup>†</sup>maieul at maieul dot net

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

<b>4 Facing pages</b>	<b>8</b>
4.1 Basic usage	8
4.2 Setting	9
4.2.1 Text width	9
4.2.2 Page number	9
4.2.3 Page breaking	9
4.2.4 Right page before <code>\Pages</code>	9
4.3 Critical and familiar footnotes	9
4.3.1 Notes height setting	10
4.3.2 Notes for one side only	10
4.3.3 Familiar notes called in the right side, but to be printed in the left side	10
<b>5 Left and right texts</b>	<b>11</b>
5.1 Environments	11
5.2 Numbering text lines and paragraphs	11
5.3 Line numbering scheme	12
5.4 Lineation system	12
5.5 Chunks	13
5.6 <code>\AtEveryPstart</code> and <code>\AtEveryPstartCall</code>	13
5.7 Language setting	13
5.8 Shifting	14
<b>6 Verse</b>	<b>14</b>
<b>7 Side notes</b>	<b>15</b>
<b>8 Parallel ledgroups</b>	<b>15</b>
8.1 General	15
8.2 Parallel ledgroups and <code>setspace</code> package	16
<b>9 Sectioning commands</b>	<b>16</b>
<b>10 Notes about page number</b>	<b>16</b>
<b>I Implementation overview</b>	<b>18</b>
<b>II Preliminaries</b>	<b>18</b>
II.1 Package's meta-data	18
II.2 Package's requirement	18
II.3 Package's options	18
II.4 Determining side and category of parallel processing	19
II.5 Text's width	20
II.6 Messages	20
<b>III Sectioning commands</b>	<b>21</b>

<b>IV Line counting</b>	<b>24</b>
IV.1 Setting lineation reset . . . . .	24
IV.2 Setting line number margin . . . . .	25
IV.3 Setting lineation start and step . . . . .	26
IV.4 Setting line flag . . . . .	27
IV.5 Setting line number style . . . . .	27
IV.6 Print marginal line number . . . . .	28
IV.7 Line-number counters and lists . . . . .	28
IV.7.1 Correspond to those in <code>reledmac</code> for regular or left text . . . . .	28
IV.7.2 Specific to <code>reledpar</code> . . . . .	29
IV.8 Reading the line-list file . . . . .	29
IV.9 Commands within the line-list file . . . . .	30
IV.10 Writing to the line-list file . . . . .	35
<b>V Marking text for notes</b>	<b>37</b>
V.1 Specific hooks and commands for notes . . . . .	37
V.1.1 Notes to be printed on one side only . . . . .	37
V.1.2 Familiar footnotes without marks . . . . .	38
V.1.3 Create hooks . . . . .	39
V.1.4 Init standards series (A,B,C,D,E,Z) . . . . .	39
<b>VI Pstart numbers dumping and restoration</b>	<b>40</b>
<b>VII Parallel environments</b>	<b>41</b>
<b>VIII Paragraph decomposition and reassembly</b>	<b>43</b>
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	43
VIII.2 Processing one line . . . . .	48
VIII.3 Line and page number computation . . . . .	52
VIII.4 Line number printing . . . . .	55
VIII.5 Pstart number printing in side . . . . .	57
VIII.6 Add insertions to the vertical list . . . . .	58
VIII.7 Penalties . . . . .	59
VIII.8 Printing leftover notes . . . . .	60
<b>IX Footnotes</b>	<b>61</b>
IX.1 Line number printing . . . . .	61
IX.2 Footnotes output specific to <code>\Pages</code> . . . . .	61
<b>X Cross referencing</b>	<b>65</b>
<b>XI Side notes</b>	<b>66</b>
<b>XII Familiar footnotes</b>	<b>67</b>
<b>XIII Verse</b>	<b>68</b>

<b>XIV Naming macros</b>	<b>70</b>
<b>XV Fixing babel and polyglossia</b>	<b>70</b>
<b>XVI Counts and boxes for parallel texts</b>	<b>72</b>
<b>XVII Checking text to be processed</b>	<b>74</b>
<b>XVIII Parallel columns</b>	<b>75</b>
<b>XIX Parallel pages</b>	<b>83</b>
XIX.1 Specific counters . . . . .	83
XIX.2 Main macro . . . . .	83
XIX.3 Ensure all notes be printed at the end of parallel pages . . . . .	89
XIX.4 Struts . . . . .	90
XIX.5 Page clearing . . . . .	90
XIX.6 Lines managing . . . . .	91
XIX.7 Page break managing . . . . .	92
XIX.8 Getting boxes content . . . . .	95
<b>XX Page numbering</b>	<b>98</b>
<b>XXI Sections' titles' commands</b>	<b>99</b>
<b>XXII Page break/no page break, depending on the specific line</b>	<b>100</b>
<b>XXIII Parallel ledgroup</b>	<b>101</b>
<b>XXIV Compatibility with eledmac</b>	<b>105</b>
<b>XXV The End</b>	<b>105</b>
<b>Appendix A Some things to do when changing version</b>	<b>106</b>
Appendix A.1 Migration to eledpar 1.4.3 . . . . .	106
Appendix A.2 Migration from eledpar to reledpar . . . . .	106
Appendix A.2.1 Deprecated options . . . . .	106
Appendix A.2.2 \renewcommandreplaced with command . . . . .	106
Appendix A.2.3 Commands the names of which have changed . . . . .	107
Appendix A.3 Migration to reledpar 2.2.0 . . . . .	107
Appendix A.4 Migration to reledmac 2.3.0 . . . . .	107
<b>References</b>	<b>107</b>
<b>Index</b>	<b>107</b>
<b>Change History</b>	<b>125</b>

# 1 Introduction

## 1.1 Aim of this package

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 `reledpar` package is an extension to `reledmac` 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. In this case, please reports them to the author via github's issues: <https://github.com/maieul/ledmac/issues/>.

This manual contains a general description of how to use `reledpar` starting in section 2; the complete source code for the package, with extensive documentation (in sections I through XXV); and an Index to the source code. As `reledpar` is an adjunct to `reledmac` we assume that you have read the `reledmac` manual. Also `reledpar` requires `reledmac` to be used, in the version distributed with version.

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. The documentation's sections are numbered in roman numeral.

On a first reading, We suggest that you should skip anything after the general documentation in first sections until I, unless you are particularly interested in the innards of `reledpar`.

## 1.2 Historical overview

Many of the code of this package is based on the `eledpar` package, which was based on the `ledpar`, created as an extension of the `ledmac` package.

Names of the package related to parallel typesetting have moved in parallel of names of the package related to critical edition.

Please read `reledmac`'s handbook in order to understand this evolution.

# 2 General

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 will want to print the text that you are editing. Unnumbered text is not printed with line numbers, and you can't use `reledmac`'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

The `reledpar` package lets you typeset two *numbered* texts in parallel<sup>1</sup>. 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

---

<sup>1</sup>You can use, anyway, `\numberlinefalse` to disable printing of line numbers.

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.

`reledmac` 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.

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

`\maxchunks`

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

```
\maxchunks{5120}
```

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

If you `\maxchunks` is too little you can get a `reledpar` 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 `\stanza`, 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, `reledmac` is a  $\TeX$  resource hog, and `reledpar` only makes things worse in this respect.

## 3 Parallel columns

### 3.1 Basic use

`pairs`

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

`\Columns`

The command `\Columns` typesets the texts in the previous pair of `Leftside` and

Rightside environments. The general scheme for parallel columns looks like this:

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

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

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

There is no required pagebreak before or after the columns.

## 3.2 Setting

### 3.2.1 Column's width

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

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

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

### 3.2.2 Column's separator

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

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

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

### 3.2.3 Column's positions

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

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

`\beforecolumnseparator`  
`\aftercolumnseparator`

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

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

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

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

If you want to revert to the previous behavior, just set with a negative value.

### 3.2.4 Mixing two columns and one column texts

`\widthliketwocolumns`

If you want to mix two-column with single-column text, you can align horizontally single-column text to two-column text with `\widthliketwocolumnstrue`. To reset this feature, use `\widthliketwocolumnsfalse`. You can also call `\widthliketwocolumns` as a global option when loading `reledmac` or `reledpar`.

`\Xnoteswidthliketwocolumns`  
`\notesXwidthliketwocolumns`

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

## 4 Facing pages

### 4.1 Basic usage

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

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

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

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



## 4.2 Setting

### 4.2.1 Text width

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

### 4.2.2 Page number

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

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

Please also read advising in 10 p. 16.

### 4.2.3 Page breaking

`\setgoalfraction` When doing parallel pages `reledpar` 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 standard value is 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. You can change it using `\setgoalfraction{<newvalue>}`.

### 4.2.4 Right page before `\Pages`

When `\Pages` are called, it starts at a new left page, in order to have parallel pages. Consequently, if it is called on a left page, it clears the current page and then lets a right void page.

`reledpar` provides two options to customize this (eventual) right page.

`prevpgstyle=<style>` in order to set the style of this page. A common value of `<style>` is `empty`. Use `prevpgstyle=empty` will suppress header and footer in this page. Please also read advising in 10 p. 16.

`prevpgnotnumbered` will make this page won’t be counted in the page counter.

## 4.3 Critical and familiar footnotes

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

### 4.3.1 Notes height setting

Since `eledpar` v1.13.0, long notes in facing pages can flow from left to right pages, and *vice-versa*.

However, the `reledmac` default setting for the maximum allotted size to notes is greater than `\textheight`. That makes impossible for long notes to flow across pages.<sup>2</sup> We have not changed this default setting, because we do not want to break compatibility with older version of `reledmac` and we want to be as close as possible to default  $\TeX$ 's feature.

So, you MUST change the default setting via `\Xmaxhnotes` (for critical notes) and `\maxhnotesX` (for familiar notes). Both commands are explained in `reledmac` handbook (6.9.4 p. 37). As an advisable setting:

```
\Xmaxhnotes{0.6\textheight}
\maxhnotesX{0.6\textheight}
```

### 4.3.2 Notes for one side only

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

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

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

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

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

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

<sup>2</sup>The same applies to  $\TeX$  normal notes. Read <http://tex.stackexchange.com/a/228283/7712> for technical informations.

```

\pend
\endnumbering
\end{Rightside}

```

## 5 Left and right texts

### 5.1 Environments

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

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

### 5.2 Numbering text lines and paragraphs

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

```

\beginnumbering
<text>
\endnumbering

```

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

The `\beginnumbering` macro resets the line number to zero, reads an auxiliary file called `<jobname>.nn` (where `<jobname>` is the name of the main input file for this job, and `nn` is 1 for the first numbered section, 2 for the second section, and so on), and then creates a new version of this auxiliary file to collect information during this run. Separate auxiliary files are maintained for right hand texts and these are named `<jobname>.nnR`, using the ‘R’ to distinguish them from the left hand and serial (non-parallel) texts.

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

```

\begin{pages}
\begin{Leftside}
\beginnumbering
...
\end{Leftside}
\begin{Rightside}
\beginnumbering
...
\end{Rightside}
\end{pages}

```

```

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

```

```

\numberstarttrue
\numberstartfalse
  \thepstartL
  \thepstartR
\skipnumbering
\hidenumbering

```

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

The command `\skipnumbering` when inserted in a line of parallel text causes the numbering of that particular line to be skipped. This can be useful if you are putting some kind of marker (even if it is only a blank line) between stanzas. Remember, parallel texts must be numbered and this provides a way to slip in an “unnumbered” line. When inserted into a numbered line the macro `\hidenumbering` causes the number for that particular line to be hidden; namely, no line number will print. Note that if you use it in `\stanza`, you must call it at the beginning of the verse.

### 5.3 Line numbering scheme

Within these environments you can designate the line numbering scheme(s) to be used.

### 5.4 Lineation system

```

\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement

```

Following `\firstlinenum{<num>}` the first line number will be `<num>`, and following `\linenumincrement{<num>}` only every `<num>`th line will have a printed number. Using these macros inside the `Leftside` and `Rightside` environments gives you independent control over the left and right numbering schemes. The `\firstsublinenum` and `\sublinenumincrement` macros correspondingly set the numbering scheme for sub-lines. 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.

```

\firstlinenum*
\linenumincrement*
\firstsublinenum*
\sublinenumincrement*
\firstlinenumR
\linenumincrementR
\firstsublinenumR
\sublinenumincrementR
\lineationR
\lineation*
\linenumberstyleR
\sublinenumberstyleR
\linenumberstyle*
\sublinenumberstyle*

```

The starred versions change both left and right numbering schemes.

The suffixed version change the right side, without regard to the position they are called.

`\lineationR` macro is the equivalent of `reledmac \lineation` macro for the right side.

`\lineation*` macro is the equivalent of `reledmac \lineation` macro for both sides.

`\linenumberstyleR` is the equivalent of `reledmac \linenumberstyle` for right text. `\sublinenumberstyleR` is the equivalent of `reledmac \sublinenumberstyle`

`\setRlineflag` right text. The starred version are for both side. A “R” is appended to the line numbers of the right texts. This may be useful for parallel columns but for parallel pages it might be more appropriate to redefine it using `\setRlineflag{flag}`. Use `\setRlineflag{}` to empty it.

## 5.5 Chunks

`\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.

`\autopar` The `\autopar` macro can be used, instead of manually inserting `\pstart... \pends`. Please read `reledmac`'s handbook (4.2.2 p. 15).

## 5.6 \AtEveryPstart and \AtEveryPstartCall

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

Consequently:

- The argument of `\AtEveryPstart` (see 4.2.4 p. 16) is called before every chunk is printed, except if you used an optional argument for the `\pstart`.
- The argument of `\AtEveryPstartCall` is called before every `\pstart`.

## 5.7 Language setting

If you are using the `babel` package or the `polyglossia` package ,with different languages (via, say, `\selectlanguage`) for the left and right texts it is particularly impor-

tant 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* language setting in a side that controls the language used in any notes for that side when they get printed. If you are using multilingual notes then it is probably safest to explicitly specify the language(s) for each note rather than relying on the language selection for the side. The right side language is also applied to the right side line numbers.

## 5.8 Shifting

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

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

## 6 Verse

If you are typesetting verses with `reledmac` 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.

`reledpar` provides an `astanza` environment which you can use instead of `\stanza`. A `astanza` environment is a chunk. Consequently left and right *verse* are matched, and not, as with standard `\stanza`, left and right *verse lines*.

Within the `astanza` environment each verse line is treated as an individual paragraph, so there must be no blank lines in the environment otherwise there will be some extraneous vertical spacing. To use `astanza`, simply replace `\stanza` by `\begin{astanza}` and add `\end{astanza}` after the ending `\&`.

The difference between `astanza` and `\stanza` is, that the letter syncs verse by verse, while the environment syncs stanza by stanza.

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.

As `astanza` is a specify type `\pstart... \pend` structure, you can:

- Add optional argument (in brackets) after `\begin{astanza}`, as the optional argument of `\pstart`.
- Use optional argument after the last `\&` as optional argument of `\pend`.

`\sethangingsymbol` Like in `reledmac`, you could use the `\sethangingsymbol` command to insert a character in each hanging line. If you use it, you must run  $\TeX$  two time. Example for the French typography

```
\sethangingsymbol{[,}
```

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

```
\sethangingsymbol{\protect\hfill}
```

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

`\thestanzaL` When using `\stanzanumtrue` (8.9 p. 41) in parallel typesetting, stanza counter is replaced by `stanzaL` counter in left side and by `stanzaR` counter in right side. Consequently, you can redefine `\thestanzaL` and `\thestanzaR` to change their aspect.

## 7 Side notes

As in `reledmac`, 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.

## 8 Parallel ledgroups

### 8.1 General

You can also make parallel ledgroups (see the documentation of `reledmac` about ledgroups, 9 p. 42). To do it you have:

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

See the following example:

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

```

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

```

## 8.2 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:

```
\setparledgroupnotespacing{\singlespacing}
```

*In effect, to have correct spacing, do not change the font size of your notes.*

## 9 Sectioning commands

The standard sectioning commands of `reledmac` are available, and provide parallel sectioning, for both two-column and two-page layout.

`\eledsectnotoc` By default, the section commands of the right side are not added to the table of contents. But you can change it, using `\eledsectnotoc{<arg>}`, where `<arg>` could be L (for left side) or R (for right side).

`\eledsectmark` By default, the headers are tokens from the left side. You can change them, using `\eledsectmark{<arg>}`, where `<arg>` could be L (for left side) or R (for right side).

## 10 Notes about page number

If you use `sameparallepagenumber` option (4.2.2 p. 9) or `prevpgnotnumbered` option (4.2.4 p. 9), please read the following paragraph if you want to manipulate page numbers manually.



In order to implement these two options, `reledpar` uses its own page counter, called `par@page`. Consequently, if you use at least one of these options:

1. If you modify `\thepage` command, use the value of `par@page` counter inside and not the value of `page` counter.
2. If you want to modify a page number, modify the value of `page` counter AND the value `par@page` counter.

Notes that `reledpar` automatically do it when you use `\frontmatter` and `\mainmatter` commands.

## I 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.

`reledmac` 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 `reledmac` code is concerned with handling this box and its contents.

`reledpar`’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.

## II Preliminaries

### II.1 Package’s meta-data

Announce the name and version of the package, which is targeted for  $\LaTeX 2\epsilon$ . The package also requires the `reledmac` package, however we do not load it automatically, because we prefer users to know it.

```

1 %<*code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledpar}[2015/09/05 v2.3.0 reledmac extension for
  parallel texts]%
4
5 %

```

### II.2 Package’s requirement

Few commands use `\xspace` command.

```

6 \RequirePackage{xspace}%
7 %

```

### II.3 Package’s options

We use `xkeyval` in order to manage options with arguments.

```

8 \RequirePackage{xkeyval}
9 %

```

With the option ‘shiftedpstarts’ a long pstart on the left side (or on the right side) does not make a blank on the corresponding pstart, but the blank is put on the bottom of the page. Consequently, the pstarts on the parallel pages are shifted, but the shift stops at every end of pages.

```
\ifshiftedpstarts 10 \newif\ifshiftedpstarts
11 \DeclareOptionX{shiftedpstarts}{\shiftedpstartstrue}
12 %
```

The `parledgroup` can be called either on `reledmac` or `reledpar`.

```
13 \DeclareOptionX{parledgroup}{\parledgrouptrue}
14 %
```

`\ifwidthliketwocolumns` The `\widthliketwocolumns` option can be called either on `reledmac` or `reledpar`.

```
15 \DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}%
16 %
```

`\ifsameparallelpagenumber` Options related to page numbering

```
\ifprevpgnotnumbered
17 \newif\ifsameparallelpagenumber
18 \newif\ifprevpgnotnumbered
19 \DeclareOptionX{sameparallelpagenumber}{\sameparallelpagenumbertrue}
20 \DeclareOptionX{prevpgnotnumbered}{\prevpgnotnumberedtrue}
21 %
```

`\prevpgstyle` We store on `\prevpgstyle` the argument of the option `prevpgstyle`.

```
22 \DeclareOptionX{prevpgstyle}{\gdef\prevpgstyle{#1}}%
23 %
```

```
24 \ProcessOptionsX%
25 %
```

## II.4 Determining side and category of parallel processing

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

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

## II.5 Text's width

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

```
\Rcolwidth
26 \newdimen\Lcolwidth
27 \Lcolwidth=0.45\textwidth
28 \newdimen\Rcolwidth
29 \Rcolwidth=0.45\textwidth
30 %
```

## II.6 Messages

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

```
\eledpar@error31 \newcommand{\reledpar@error}[2]{\PackageError{reledpar}{#1}{#2}}
32 %
```

```
\led@err@TooManyPstarts33 \newcommand*\led@err@TooManyPstarts}{%
34 \reledpar@error{Too many \string\pstart\space without printing.
35 Some text will be lost}{\@ehc}}
36 %
```

```
\led@err@BadLeftRightPstarts37 \newcommand*\led@err@BadLeftRightPstarts}[2]{%
38 \reledpar@error{The numbers of left (#1) and right (#2)
39 \string\pstart s do not match}{\@ehc}}
40 %
```

```
\led@err@LeftOnRightPage41 \newcommand*\led@err@LeftOnRightPage}{%
\led@err@RightOnLeftPage42 \reledpar@error{The left page has ended on a right page}{\@ehc}}
43 \newcommand*\led@err@RightOnLeftPage}{%
44 \reledpar@error{The right page has ended on a left page}{\@ehc}}
45 %
```

```
\led@err@Leftside@PreviousNotPrinted46 \newcommand*\led@err@Leftside@PreviousNotPrinted}{%
\led@err@Rightside@PreviousNotPrinted47 \reledpar@error{You call a new Leftside environment while the previous
one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
48 \newcommand*\led@err@Rightside@PreviousNotPrinted}{%
49 \reledpar@error{You call a new Rightside environment while the previous
one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
50 %
```

```

\led@err@Pages@InsideEnv 51 \newcommand*\led@err@Pages@InsideEnv}{%
\led@err@Columns@InsideEnv 52 \reledpar@error{\string\Pages\space must be called *outside* of the `
                             pages` environment}{\@ehc}}
53 \newcommand*\led@err@Columns@InsideEnv}{%
54 \reledpar@error{\string\Columns\space must be called *outside* of the `
                             pairs` environment}{\@ehc}}
55 %

\error@fail@patch@thepage 56 \newcommand{\led@error@fail@patch@thepage}{%
57 \reledpar@error{Fail to patch \string\@thepage\space command.}{\@ehc}%
58 }%
59 %

\fail@patch@pagenumbering 60 \newcommand{\led@error@fail@patch@pagenumbering}{%
61 \reledpar@error{Fail to patch \string\pagenumbering\space command.}{\@ehc
62 }%
63 }%
64 %

\error@fail@patch@@mempnum 64 \newcommand{\led@error@fail@patch@@mempnum}{%
65 \reledpar@error{Fail to patch \string\@mempnum\space command.}{\@ehc}%
66 }%
67 %

\error@fail@patch@@outputpage 68 \newcommand{\led@error@fail@patch@@outputpage}{%
69 \reledpar@error{Fail to patch \string\@outputpage\space command.}{\@ehc}%
70 }%
71 %

```

### III 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.

```

72 \newcount\section@numR
73 \section@numR=\z@
74 %

```

`\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 `reledmac`.

```

75 \pst@rtedLfalse
76 \newif\ifpst@rtedR
77
78 %

```

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

```

79 \newcommand*{\beginnumberingR}{%
80   \ifnumberingR
81     \led@err@NumberingStarted
82     \endnumberingR
83   \fi
84   \global\l@dnumpstartsR \z@
85   \global\pst@rtedRfalse
86   \global\numberingRtrue
87   \global\advance\section@numR \@ne
88   \global\absline@numR \z@
89   \gdef\normal@page@breakR{}
90   \gdef\l@prev@pbR{}
91   \gdef\l@prev@nopbR{}
92   \global\line@numR \z@
93   \global\@lockR \z@
94   \global\sub@lockR \z@
95   \global\sublines@false
96   \global\let\next@page@numR\relax
97   \global\let\sub@change\relax
98   \message{Section \the\section@numR R }%
99   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
100  \l@dend@stuff
101  \setcounter{pstartR}{1}
102  \begingroup
103  \initnumbering@sectcountR
104  \gdef\eled@sectionsR@{}%
105  \if@noeled@sec\else%
106    \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@numR R
107  }{}{\makeatother%
108    \immediate\openout\eled@sectioningR@out=\jobname.eledsec\the\
109  section@numR R\relax%
110  \fi%
111  }
112  %

```

**\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 `reledmac`.

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

```

111 \def\endnumberingR{%
112   \ifnumberingR
113     \global\numberingRfalse
114     \normal@pars
115     \ifnum\l@dnumpstartsR=0%
116       \led@err@NumberingWithoutPstart%
117     \fi%
118     \ifl@dpairing
119       \global\pst@rtedRfalse
120     \else
121       \ifx\insertlines@listR\empty\else
122         \global\noteschanged@true
123       \fi
124       \ifx\line@listR\empty\else
125         \global\noteschanged@true
126       \fi
127     \fi
128     \ifnoteschanged@
129       \led@mess@NotesChanged
130     \fi
131   \else
132     \led@err@NumberingNotStarted
133   \fi
134 \endgroup
135 \if@noeled@sec\else%
136   \immediate\closeout\eled@sectioningR@out%
137 \fi%
138 }
139
140 %

```

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

```

141 \newcounter{chapterR}
142 \newcounter{sectionR}
143 \newcounter{subsectionR}
144 \newcounter{subsubsectionR}
145 \newcommand{\initnumbering@sectcountR}{
146   \let\c@chapter\c@chapterR
147   \let\c@section\c@sectionR
148   \let\c@subsection\c@subsectionR
149   \let\c@subsubsection\c@subsubsectionR
150 }
151 %

```

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

```

\resumenumberingR
152 \newcommand*{\pausenumberingR}{%
153   \endnumberingR\global\numberingRtrue}

```

```

154 \newcommand*{\resumenumberingR}{%
155   \ifnumberingR
156     \global\pst@rtedRtrue
157     \global\advance\section@numR \@ne
158     \led@mess@SectionContinued{\the\section@numR R}%
159     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
160     \l@dend@stuff
161     \begingroup%
162     \initnumbering@sectcountR%
163   \else
164     \led@err@numberingShouldHaveStarted
165     \endnumberingR
166     \beginnumberingR
167   \fi}
168
169 %

```

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

```

170 \newcommand*{\memorydumpL}{%
171   \endnumbering
172   \numberingtrue
173   \global\pst@rtedLtrue
174   \global\advance\section@num \@ne
175   \led@mess@SectionContinued{\the\section@num}%
176   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
177   \l@dend@stuff}
178
179 \newcommand*{\memorydumpR}{%
180   \endnumberingR
181   \numberingRtrue
182   \global\pst@rtedRtrue
183   \global\advance\section@numR \@ne
184   \led@mess@SectionContinued{\the\section@numR R}%
185   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
186   \l@dend@stuff}
187
188 %

```

## IV Line counting

### IV.1 Setting lineation reset

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. `reledpar` lets you choose different schemes for the left and right texts.



`\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`.

```

189 \newcommand*{\lineationR}[1]{%
190   \ifnumbering
191     \led@err@LineationInNumbered
192   \else
193     \def\@tempa{#1}\def\@tempb{page}%
194     \ifx\@tempa\@tempb
195       \global\bypage@Rtrue
196       \global\bypstart@Rfalse
197       \unless\ifnocritical@%
198         \Xpstart[] [false]%
199       \fi%
200     \else
201       \def\@tempb{pstart}%
202       \ifx\@tempa\@tempb
203         \global\bypage@Rfalse
204         \global\bypstart@Rtrue
205         \unless\ifnocritical@%
206           \Xpstart%
207         \fi%
208       \else
209         \def\@tempb{section}
210         \ifx\@tempa\@tempb
211           \global\bypage@Rfalse%
212           \global\bypstart@Rfalse%
213           \unless\ifnocritical@%
214             \Xpstart[] [false]%
215           \fi%
216         \else
217           \led@warn@BadLineation
218         \fi%
219       \fi
220     \fi
221   \fi}}
222 %

```

`\lineation*` `\lineation*` change the lineation system for both sides.

```

223 \WithSuffix\newcommand\lineation*[1]{%
224   \lineation{#1}%
225   \lineationR{#1}%
226 }%
227 %

```

## IV.2 Setting line number margin

`\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

`\line@marginR`

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 would like; if it is 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.

It is defined only once time, in `reledmac`.

```
228 \newcount\line@marginR
229 %
```

By default put right text numbers at the right.

```
230 \line@marginR=\@ne
231
232 %
```

### IV.3 Setting lineation start and step

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

```
233 \newcounter{firstlinenumR}
234 \setcounter{firstlinenumR}{5}
235 \newcounter{linenumincrementR}
236 \setcounter{linenumincrementR}{5}
237 %
```

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

```
238 \newcounter{firstsublinenumR}
239 \setcounter{firstsublinenumR}{5}
240 \newcounter{sublinenumincrementR}
241 \setcounter{sublinenumincrementR}{5}
242
243 %
```

`\firstlinenum` `\linenumincrement` `\firstsublinenum` `\sublinenumincrement` These are the user's macros for changing (sub) line numbers. They are defined in `reledmac`. The starred versions are specific to `eledpar`.

```
244 \WithSuffix\newcommand\firstlinenum*[1]{%
\sublinenumincrement \setcounter{firstlinenumR}{#1}%
\firstlinenum* \setcounter{firstlinenum}{#1}%
\linenumincrement* }
\firstsublinenum* \WithSuffix\newcommand\linenumincrement*[1]{%
\sublinenumincrement* \setcounter{linenumincrementR}{#1}%
```

```

250 \setcounter{linenumincrement}{#1}%
251 }
252 \WithSuffix\newcommand\firstsublinenum*[1]{%
253 \setcounter{subfirstlinenumR}{#1}%
254 \setcounter{subfirstlinenum}{#1}%
255 }
256 \WithSuffix\newcommand\sublinenumincrement*[1]{%
257 \setcounter{sublinenumincrementR}{#1}%
258 \setcounter{sublinenumincrement}{#1}%
259 }
260 %

```

`\firstlinenumR` And the ‘R’ suffixed version.

```

\linenumincrementR
\firstsublinenumR
\sublinenumincrementR
261 \newcommand\firstlinenumR[1]{%
262 \setcounter{firstlinenumR}{#1}%
263 }
264 \newcommand\linenumincrementR[1]{%
265 \setcounter{linenumincrementR}{#1}%
266 }
267 \newcommand\firstsublinenumR[1]{%
268 \setcounter{subfirstlinenumR}{#1}%
269 }
270 \newcommand\sublinenumincrementR[1]{%
271 \setcounter{sublinenumincrementR}{#1}%
272 }
273 %

```

#### IV.4 Setting line flag

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

```

274 \newcommand{\setRlineflag}[1]{%
275 \gdef\Rlineflag{#1}%
276 }
277 \setRlineflag{R}
278 %

```

#### IV.5 Setting line number style

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

```

279 \newcommand*\linenumrepr[1]{\@arabic{#1}}
280 \newcommand*\sublinenumrepr[1]{\@arabic{#1}}
281
282 %

```

`\linenumberstyleR` The style can be changed by some user level command  
`\sublinenumberstyleR`

```

283 \newcommand*\linenumberstyleR}[1]{%
284   \def\linenumrepR##1{\@nameuse{##1}}
285 \newcommand*\sublinenumberstyleR}[1]{%
286   \def\sublinenumrepR##1{\@nameuse{##1}}
287 %

```

`\linenumberstyle*` And for both side.  
`\sublinenumberstyle*`

```

288 \WithSuffix\newcommand\linenumberstyle*[1]{%
289   \linenumberstyle{##1}%
290   \linenumberstyleR{##1}%
291 }%
292
293 \WithSuffix\newcommand\sublinenumberstyle*[1]{%
294   \sublinenumberstyle{##1}%
295   \sublinenumberstyleR{##1}%
296 }%
297 %
298 %

```

## IV.6 Print marginal line number

`\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`.

```

299 \newcommand*\leftlinenumR}{%
300   \l@dlinenumR
301   \kern\linenumsep}
302 \newcommand*\rightlinenumR}{%
303   \kern\linenumsep
304   \l@dlinenumR}
305 \newcommand*\l@dlinenumR}{%
306   \numlabfont\linenumrepR{\line@numR}\@Rlineflag%
307   \ifsublines@
308     \ifnum\subline@num>\z@
309       \unskip\fullstop\sublinenumrepR{\subline@numR}%
310     \fi
311   \fi}
312
313 %

```

## IV.7 Line-number counters and lists

### IV.7.1 Correspond to those in `reledmac` for regular or left text

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

`\line@numR` The count `\line@numR` stores the line number that is 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 have actually printed, no matter what numbers we attached to them.

```
314 \newcount\line@numR
315 \newcount\subline@numR
316 \newcount\absline@numR
317
318 %
```

`\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 `reledmac` manual.

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

```
319 \list@create{\line@listR}
320 \list@create{\insertlines@listR}
321 \list@create{\actionlines@listR}
322 \list@create{\actions@listR}
323
324 %
```

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

```
325 \newcount\page@numR
326
327 %
```

#### IV.7.2 Specific to `reledpar`

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

```
328 \list@create{\linesinpar@listL}
329 \list@create{\linesinpar@listR}
330 \list@create{\maxlinesinpar@list}
331
332 %
```

### IV.8 Reading the line-list file

`\list@clearing@regR` \Clear the right lines for `\read@linelist`

```
333 \newcommand{\list@clearing@regR}{%
334   \list@clear{\line@listR}%
335   \list@clear{\insertlines@listR}%
```

```

336 \list@clear{\actionlines@listR}%
337 \list@clear{\actions@listR}%
338 \list@clear{\linesinpar@listR}%
339 \list@clear{\linesonpage@listR}
340 }
341 %

```

`\read@linelist` `\read@linelist{⟨file⟩}` is the control sequence that is called by `\beginnumbering` (via `\line@list@stuff`) to open and process a line-list file; its argument is the name of the file. . It is defined only once time in `reledmac`.

## IV.9 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@regR` is called by `\@nl` if we are on a right side. It does everything related to the start of a new line of numbered text on a right side.

```

342 \newcommand{\@nl@regR}{%
343   \ifx\l@dchset@num\relax \else
344     \advance\absline@numR \@ne
345     \set@line@action
346     \let\l@dchset@num\relax
347     \advance\absline@numR \m@ne
348     \advance\line@numR \m@ne% % do we need this?
349   \fi
350   \advance\absline@numR \@ne
351   \ifx\next@page@numR\relax \else
352     \page@action
353     \let\next@page@numR\relax
354   \fi
355   \ifx\sub@change\relax \else
356     \ifnum\sub@change>\z@
357       \sublines@true
358     \else
359       \sublines@false
360     \fi
361     \sub@action
362     \let\sub@change\relax
363   \fi
364   \ifcase\@lockR
365   \or
366     \@lockR \tw@
367   \or\or
368     \@lockR \z@

```

```

369 \fi
370 \ifcase\sub@lockR
371 \or
372 \sub@lockR \tw@
373 \or\or
374 \sub@lockR \z@
375 \fi
376 \ifsublines@
377 \ifnum\sub@lockR<\tw@
378 \advance\subline@numR \@ne
379 \fi
380 \else
381 \ifnum\@lockR<\tw@
382 \advance\line@numR \@ne \subline@numR \z@
383 \fi
384 \fi}
385
386
387 %

```

`\last@page@numR` `\last@page@numR` store the page number of the last right page. It is modified by `\fix@page` `\fix@page`, defined by `reledmac`.

```

388 \newcount\last@page@numR
389 \last@page@numR=-10000
390
391 %

```

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

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

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

`\page@action` `\page@action` adds an entry to the action-code list to change the page number. It is defined in `reledmac`.

`\set@line@action` `\set@line@action` adds an entry to the action-code list to change the visible line number. It is defined in `reledmac`.

`\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. It is defined in `reledmac`.

`\do@lockon` `\lock@on` adds an entry to the action-code list to turn line number locking on. The `\do@lockonR` current setting of the sub-lineation flag tells us whether this applies to line numbers or sub-line numbers. It is defined in `reledmac`, however the code specific to right side is defined here, in `\do@lockonR`.

```

392 \newcount\@lockR
393 \newcount\sub@lockR
394
395 \newcommand*\do@lockonR}{%
396   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
397   \ifsublines@
398     \xright@appenditem{-1005}\to\actions@listR
399     \ifnum\sub@lockR=\z@
400       \sub@lockR \@ne
401     \else
402       \ifnum\sub@lockR=\thr@@
403         \sub@lockR \@ne
404       \fi
405     \fi
406   \else
407     \xright@appenditem{-1003}\to\actions@listR
408     \ifnum\@lockR=\z@
409       \@lockR \@ne
410     \else
411       \ifnum\@lockR=\thr@@
412         \@lockR \@ne
413       \fi
414     \fi
415   \fi}
416
417 %

```

`\lock@off` `\lock@off` adds an entry to the action-code list to turn line number locking off. It is defined in `reledmac`, however the code specific to right side is defined here, in `\do@lockoffR` `\do@lockoffR` `\do@lockoffR`. `\skip@lockoff`

```

418
419
420 \newcommand*\do@lockoffR}{%
421   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
422   \ifsublines@
423     \xright@appenditem{-1006}\to\actions@listR
424     \ifnum\sub@lockR=\tw@
425       \sub@lockR \thr@@
426     \else
427       \sub@lockR \z@
428     \fi
429   \else
430     \xright@appenditem{-1004}\to\actions@listR
431     \ifnum\@lockR=\tw@

```



```

432 \@lockR \thr@@
433 \else
434 \@lockR \z@
435 \fi
436 \fi}
437
438
439 %

```

`\n@num`

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

`\@ref@regR`

`\insert@countR`

- #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`.

```

440 \newcount\insert@countR
441 %

```

- #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`.

`\@ref` itself is defined in `reledmac`. It calls `\ref@reg` or `\ref@regR`, depending whether we are in left or right side. Here, we define only `\ref@regR`, `\ref@reg` is already defined in `reledmac`.

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

```

442 \newcommand*{\@ref@regR}[2]{%
443 \global\advance\@edtext@level by 1%
444 \global\insert@countR=#1\relax
445 \loop\ifnum\insert@countR>\z@
446 \xright@appenditem{\the\absline@numR}\to\insertlines@listR
447 \global\advance\insert@countR \m@ne
448 \repeat
449 %

```

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.

```

450 \begingroup
451 \let\@ref=\dummy@ref
452 \let\@lopR\@gobble
453 \let\page@action=\relax
454 \let\sub@action=\relax

```

```

455 \let\set@line@action=\relax
456 \let\@lab=\relax
457 \let\@lemma=\relax
458 \let\@sw\@gobblethree%
459 #2
460 \global\endpage@num=\page@numR
461 \global\endline@num=\line@numR
462 \global\endsubline@num=\subline@numR
463 \endgroup
464 %

```

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

```

465 \xright@appenditem%
466   {\the\page@numR|\the\line@numR|}%
467   \ifsublines@ \the\subline@numR \else 0\fi|}%
468   \the\endpage@num|\the\endline@num|}%
469   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
470 %

```

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

```

471 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
@edtext@level\endcsname}%
472 %

```

Declare and init boolean for lemma in this level.

```

473 \providebool{lemmacommand@\the\@edtext@level}%
474 \boolfalse{lemmacommand@\the\@edtext@level}%
475 %

```

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

```

476 #2
477 % Now, we store the list of \protect\cs{@sw} of this current \protect\cs{
edtext} as an element of
478 % the global list of list of \protect\cs{@sw} for a \protect\cs{edtext}
depth.
479 % \begin{macrocode}
480 \ifnum\@edtext@level>0%
481 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtextR@\the\@edtext@level\endcsname}}%
482 \ifcsundef{sw@list@edtextR@\the\@edtext@level}{\
create@this@edtext@level}{}%
483 \letcs{\@tmp}{sw@list@edtextR@\the\@edtext@level}%
484 \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}%
485 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
486 \global\cslet{sw@list@edtextR@\the\@edtext@level}{\@tmp}%
487 \fi%
488 %

```

Decrease edtext level counter.

```
489     \global\advance\@edtext@level by -1%
490 }
491 %
```

`\@pend` `\@pend{<num>}` adds its argument to the `\linesinpar@listL` list, and analogously for `\@pendR`. If needed, it resets line number. Both are defined in `reledmac`, but they are empty. They are really defined only in `reledpar`.

```
492 \renewcommand*\@pend}[1]{%
493   \ifbypstart@global\line@num=0\fi%
494   \xright@appenditem{#1}\to\linesinpar@listL}
495 \renewcommand*\@pendR}[1]{%
496   \ifbypstart@R@global\line@numR=0\fi
497   \xright@appenditem{#1}\to\linesinpar@listR}
498 %
499 %
```

`\@lopL` `\@lopL{<num>}` adds its argument to the `\linesonpage@listL` list, and analogously for `\@lopR`. Both are defined in `reledmac`, but they are empty. They are really defined only in `reledpar`.

```
500 \renewcommand*\@lopL}[1]{%
501   \xright@appenditem{#1}\to\linesonpage@listL}
502 \renewcommand*\@lopR}[1]{%
503   \xright@appenditem{#1}\to\linesonpage@listR}
504 %
505 %
```

## IV.10 Writing to the line-list file

We have now defined all the counters, lists, and commands involved in reading the line-list file at the start of a section. Now we will cover the commands that `reledmac` 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`.

```
506 \newwrite\linenum@outR
507 %
```

`\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
508 \newif\iffirst@linenum@out@R
509   \first@linenum@out@Rtrue
510 %
```

`\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.

```

511 \newcommand*\line@list@stuffR}[1]{%
512   \read@linelist{#1}%
513   \iffirst@linenum@out@R
514     \immediate\closeout\linenum@outR
515     \global\first@linenum@out@Rfalse
516     \immediate\openout\linenum@outR=#1
517     \immediate\write\linenum@outR{\string\line@list@version{\
this@line@list@version}}%
518   \else
519     \if@minipage%
520       \leavevmode%
521     \fi%
522     \closeout\linenum@outR%
523     \openout\linenum@outR=#1%
524   \fi}
525
526 %

```

`\new@lineL` The `\new@lineL` macro sends the `\@n1` command to the left text line-list file, to mark the start of a new text line.

```

527 \newcommand*\new@lineL}{%
528   \write\linenum@out{\string\@n1[\the\c@page][\thepage]}%
529 %

```

`\new@lineR` The `\new@lineR` macro sends the `\@n1` command to the right text line-list file, to mark the start of a new text line.

```

530 \newcommand*\new@lineR}{%
531   \write\linenum@outR{\string\@n1[\the\c@page][\thepage]}%
532 %

```

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

`\flag@end`

`\startsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate instructions to the line-list file. There are both defined in `reledmac`.

`\endsub`

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

`\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. It is defined in `reledmac`.

`\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. It is defined in `reledmac`.

`\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. They are defined in `reledmac`.

`\endlock`

`\skipnumbering`

## V Marking text for notes

The `\edtext` 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`

`\edtext`

`\set@line`

The `\set@line` macro is called by `\edtext` to put the line-reference field and font specifier for the current block of text into `\l@d@nums`. It is defined in `reledmac`.

### V.1 Specific hooks and commands for notes

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

```
\newseries@par:33 \newcommand{\newseries@par}[1]{%
534 %
```

#### V.1.1 Notes to be printed on one side only

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

```
535 \unless\ifnofamiliar@%
536 \csgdef{onlysideX@#1}{}%
537 \global\newbool{keepforsideX@#1}%
538 \fi%
539 \unless\ifnocritical@%
540 \global\newbool{keepforXside@#1}%
541 \csgdef{Xonlyside@#1}{}%
542 \fi%
543 %
```

### V.1.2 Familiar footnotes without marks

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

First, check the `nofamiliar` option of `reledmac`.

```
544 \unless\ifnofamiliar@%
545 % So declare the list.
546 % \begin{macrocode}
547 \expandafter\list@create\csname footnote#1@mk\endcsname%
548 %
```

Then, declare the `\footnoteXnomk` command.

```
549 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
550 %
```

First step: just call the normal `\footnoteX`, saying that we do not want to print the mark.

```
551 \toggletrue{nomk@}%
552 \csuse{footnote#1}{##1}%
553 \togglefalse{nomk@}%
554 %
```

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

```
555 \letcs{\@tmp}{footnote#1@mk}%
556 \numdef\@tmpa{\csuse{c@footnote#1}}%
557 \global\xright@appenditem{\@tmpa}\to\@tmp%
558 \global\cslet{footnote#1@mk}{\@tmp}%
559 }%
560 %
```

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

```
561 \expandafter\newcommand\csname footnote#1mk\endcsname{%
562 %
```

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

```
563 \letcs{\@tmp}{footnote#1@mk}%
564 \gl@p\@tmp\to\@tmpa%
565 \global\cslet{footnote#1@mk}{\@tmp}%
566 %
```

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

```
567 \letcs{\old@footnote}{c@footnote#1}%
568 \setcounter{footnote#1}{\@tmpa}%
569 %
```

Define the footnote mark and print it

```
570     \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
571     \csuse{@footnotemark#1}%
572 %
```

Restore previous footnote counter and finally add space.

```
573     \setcounter{footnote#1}{\old@footnote}%
574     \xspace%
575 }%
576 %
```

End of tools for familiar notes without marks

```
577 \fi
578 %
```

End of `\newseries@par`.

```
579 }%
580 %
```

### V.1.3 Create hooks

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

```
581 \unless\ifnocritical@%
582   \newhookcommand@series{Xonlyside}%
583 \fi%
584 \unless\ifnofamiliar@%
585   \newhookcommand@series{onlysideX}%
586 \fi
587
588
589 %
```

### V.1.4 Init standards series (A,B,C,D,E,Z)

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

```
590 \newcommand{\init@series@par}{%
591   \def\do##1{\newseries@par{##1}}%
592   \dolistloop{\@series}%
593 }%
594 \init@series@par%
595 %
```

## VI Pstart numbers dumping and restoration

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

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

```
\list@pstartL@pc96 \list@create{\list@pstartL@pc}%
\list@pstartR@pc97 \list@create{\list@pstartR@pc}%
98 %
```

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

```
\dump@pstartL@pc99 \def\dump@pstartL@pc{%
\dump@pstartR@pc00 \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
01 \global\cslet{numberpstart@L\the\l@dnumpstartsL}{\ifnumberpstart}%
02 }%
03
04 \def\dump@pstartR@pc{%
05 \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
06 \global\cslet{numberpstart@R\the\l@dnumpstartsR}{\ifnumberpstart}%
07 }%
08
09 %
```

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

```
\restore@pstartR@pc
10 \def\restore@pstartL@pc{%
11 \ifx\list@pstartL@pc\empty\else%
12 \gl@p\list@pstartL@pc\to\@temp%
13 \global\c@pstartL=\@temp%
14 \fi%
15 }%
16 \def\restore@pstartR@pc{%
17 \ifx\list@pstartR@pc\empty\else%
18 \gl@p\list@pstartR@pc\to\@temp%
19 \global\c@pstartR=\@temp%
20 \fi%
21 }%
22 %
```



## VII Parallel environments

The initial set up for parallel processing is deceptively simple.

`pairs` `pages`

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

```

623 \newenvironment{pairs}{%
624   \l@dpairingtrue
625   \l@dpagingfalse
626   \initnumbering@quote
627   \at@begin@pairs%
628 }{%
629   \l@dpairingfalse
630 }
631
632 %

```

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

```

633 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
634 \def\at@begin@pairs{}%
635
636 %

```

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.

```

637 \newenvironment{pages}{%
638   \l@dpairingtrue
639   \l@dpagingtrue
640   \initnumbering@quote
641   \setlength{\Lcolwidth}{\textwidth}%
642   \setlength{\Rcolwidth}{\textwidth}%
643 }{%
644   \l@dpairingfalse
645   \l@dpagingfalse
646 }
647
648 %

```

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

```

649 \newif\ifinstanzaL
650 \newif\ifinstanzaR
651 %

```

**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.

```

652 \newenvironment{Leftside}{%
653   \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
654     \led@err@Leftside@PreviousNotPrinted%
655   \fi%
656   \ledRcolfalse
657   \setcounter{pstartL}{1}
658   \let\pstart\pstartL
659   \let\thepstart\thepstartL
660   \let\pend\pendL
661   \let\memorydump\memorydumpL
662   \Leftsidehook
663   \let\old@startstanza\@startstanza
664   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
665 }{
666   \Leftsidehookend}
667 %

```

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

```

668 \newcommand*\Leftsidehook{}
669 \newcommand*\Leftsidehookend{}
670 \newcommand*\Rightsidehook{}
671 \newcommand*\Rightsidehookend{}
672
673 %

```

**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.

```

674 \newenvironment{Rightside}{%
675   \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
676     \led@err@Rightside@PreviousNotPrinted%
677   \fi%
678   \ledRcoltrue
679   \let\beginnumbering\beginnumberingR
680   \let\endnumbering\endnumberingR
681   \let\pausenumbering\pausenumberingR
682   \let\resumenumbering\resumenumberingR
683   \let\memorydump\memorydumpR
684   \let\thepstart\thepstartR
685   \let\pstart\pstartR
686   \let\pend\pendR
687   \let\ledpb\ledpbR

```

```

688 \let\lednopb\lednopbR
689 \let\lineation\lineationR
690 \Rightsidehook
691 \let\old@startstanza\@startstanza
692 \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
693 }{%
694 \ledRcolfalse
695 \Rightsidehookend
696 }
697
698 %

```

## VIII 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.

### VIII.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 is 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.

```

699 \newcount\num@linesR
700 \newbox\one@lineR
701 \newcount\par@lineR
702 %

```

`\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 is to be numbered.

`\pstartR`

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

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

```

703
704 \newcounter{pstartL}
705 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
706 \newcounter{pstartR}
707 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
708
709 \newcommandx*{\pstartL}[1][1]{%
710   \if@nobreak%
711     \let\@oldnobreak\@nobreaktrue%
712   \else%
713     \let\@oldnobreak\@nobreakfalse%
714   \fi%
715   \@nobreaktrue%
716   \ifluatex%
717     \xdef\l@luatextextdir@L{\the\textdir}%
718     \xdef\l@luatexpardir@L{\the\pardir}%
719     \xdef\l@luatexbodydir@L{\the\bodydir}%
720   \fi%
721   \ifnumbering \else%
722     \led@err@PstartNotNumbered%
723     \beginnumbering%
724   \fi%
725   \ifnumberedpar@%
726     \led@err@PstartInPstart%
727     \pend%
728   \fi%
729 %

```

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

```

730 \ifpst@rtedL\else%
731   \list@clear{\inserts@list}%
732   \global\let\next@insert=\empty%
733   \global\pst@rtedLtrue%
734 \fi%
735 \begingroup\normal@pars%
736 %

```

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

```

737 \global\advance\l@dnumpstartsL \@one%
738 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
739   \led@err@TooManyPstarts%
740   \global\l@dnumpstartsL=\l@dc@maxchunks%
741 \fi%
742 \global\setnamebox{l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup%
743 %

```

We set all the usual interline penalties to zero; this ensures that there will 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.

```

744 \l@dzeropenalties%
745 \ifautopar\else%
746   \ifnumberpstart%
747     \ifsidepstartnum%
748       \else%
749         \thepstartL%
750       \fi%
751     \fi%
752   \fi%
753 \hsize=Lcolwidth%
754 \numberedpar@true%
755 \iflabelpstart\protected@edef\@currentlabel%
756   {\p@pstartL\thepstartL}\fi%
757 %

```

Dump the optional arguments

```

758 \ifstrempty{#1}%
759   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\at@every@pstart}}%
760   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent#1}}%
761   \at@every@pstart@call%
762 %

```

Gobble following space (automatically done if there is no optional argument)

```

763 \ignorespaces%
764 %
765 }
766 %

```

The same for right side.

```

767 \newcommandx*{\pstartR}[1][1]{%
768   \if@nobreak%
769     \let\@oldnobreak\@nobreaktrue%
770   \else%
771     \let\@oldnobreak\@nobreakfalse%
772   \fi%
773   \@nobreaktrue%
774 \ifluatex%
775   \xdef\l@luatextextdir@R{\the\textdir}%
776   \xdef\l@luatexpardir@R{\the\pardir}%
777   \xdef\l@luatexbodydir@R{\the\bodydir}%
778 \fi%
779 \ifnumberingR \else%
780   \led@err@PstartNotNumbered%
781   \beginnumberingR%
782 \fi%
783 \ifnumberedpar@%
784   \led@err@PstartInPstart%

```

```

785 \pendR%
786 \fi%
787 \ifpstrtedR\else%
788 \listclear{\inserts@listR}%
789 \global\let\next@insertR=\empty%
790 \global\pstrtedRtrue%
791 \fi%
792 \begingroup\normal@pars%
793 \global\advance\l@dnumstartsR \@ne%
794 \ifnum\l@dnumstartsR>\l@dc@maxchunks%
795 \led@err@TooManyPstarts%
796 \global\l@dnumstartsR=\l@dc@maxchunks%
797 \fi%
798 \global\setnamebox{\l@dc@rawbox\the\l@dnumstartsR}=\vbox\bgroup%
799 \l@dzeropenalties%
800 \ifautopar\else%
801 \ifnumberpstart%
802 \ifsidepstartnum\else%
803 \thepstartR%
804 \fi%
805 \fi%
806 \fi%
807 \hsize=\Rcolwidth%
808 \numberedpar@true%
809 \iflabelpstart\protected@edef\@currentlabel%
810 {\pstartR\thepstartR}\fi%
811 \ifstrempy{#1}%
812 {\csgdef{before@pstartR@the\l@dnumstartsR}{\at@every@pstart}}%
813 {\csgdef{before@pstartR@the\l@dnumstartsR}{\noindent#1}}%
814 \at@every@pstart@call%
815 \ignorespaces%
816 }
817 %

```

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

```

818 \newcommand*{\pendL}[1][1]{%
819 \ifnumbering \else%
820 \led@err@PendNotNumbered%
821 \fi%
822 \ifnumberedpar@ \else%
823 \led@err@PendNoPstart%
824 \fi%
825 %

```

We immediately call `\endgraf` to end the paragraph; this ensures that there will be no large interline penalties to prevent us from slicing the paragraph into pieces.

```

826 \endgraf\global\num@lines=\prevgraf\egroup%
827 \global\par@line=0%

```

```

828 %
      End the group that was begun in the \pstart.

829 \endgroup%
830 \ignorespaces%
831 \@oldnbreak%
832 \dump@pstartL@pc%
833 \ifnumberpstart%
834   \addtocounter{pstartL}{1}%
835 \fi
836 \parledgroup@beforenotes@save{L}%
837 %

      Dump content of the optional argument.

838 \ifstrempy{#1}%
839   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
840   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
841 }
842 %

```

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

```

843 \newcommand*{\pendR}[1][1]{%
844   \ifnumberingR \else%
845     \led@err@PendNotNumbered%
846   \fi%
847   \ifnumberedpar@ \else%
848     \led@err@PendNoPstart%
849   \fi%
850   \endgraf\global\num@linesR=\prevgraf\egroup%
851   \global\par@lineR=0%
852   \endgroup%
853   \ignorespaces%
854   \@oldnbreak%
855   \dump@pstartR@pc%
856   \ifnumberpstart%
857     \addtocounter{pstartR}{1}%
858   \fi%
859   \parledgroup@beforenotes@save{R}%
860   \ifstrempy{#1}%
861     {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
862     {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
863 }
864 %
865 %

```

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

```

866 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call{#1}}%
867 \gdef\at@every@pstart@call{}%
868 %

```

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

`\ifprint@last@after@pendR`

```

869 \newif\ifprint@last@after@pendL%
870 \newif\ifprint@last@after@pendR%
871 %

```

## VIII.2 Processing one line

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

`\l@dleftbox` A line of left text will be put in the box `\l@dleftbox`, and analogously for a line of right text.

`\l@drightbox`

```

872 \newbox\l@dleftbox
873 \newbox\l@drightbox
874
875 %

```

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

`\countRline`

```

876 \newcount\countLline
877 \countLline \z@
878 \newcount\countRline
879 \countRline \z@
880
881 %

```

`\@donereallinesL` We need to know the number of ‘real’ lines output (i.e., those that have been input by the user), and the total lines output (which includes any blank lines output for synchronisation).

`\@donetotallinesL`

`\@donereallinesR`

`\@donetotallinesR`

```

882 \newcount\@donereallinesL
883 \newcount\@donetotallinesL
884 \newcount\@donereallinesR
885 \newcount\@donetotallinesR
886
887 %

```

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

```

888 \newcommand*\do@lineL{%
889 \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscL}%
890 \advance\countLline \@ne%

```



```

891 \ifvbox\namebox{l@dLcolrawbox\the\l@dpscL}%
892 {\vbadness=10000%
893 \splittopskip=\z@%
894 \do@lineLhook%
895 \l@emptyd@ta%
896 \global\setbox\one@line=\vsplit\namebox{l@dLcolrawbox\the\l@dpscL}%
897 to\baselineskip}%
898 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startL
}{}%
899 \unvbox\one@line \global\setbox\one@line=\lastbox%
900 \getline@numL%
901 \ifnum\@lock>\@ne%
902 \inserthangingsymboltrue%
903 \else%
904 \inserthangingsymbolfalse%
905 \fi%
906 \setbox\l@dleftbox%
907 \hb@xt@ \Lcolwidth{%
908 \ifl@dhidenumber%
909 \global\l@dhidenumberfalse%
910 \f@x@l@cks%
911 \else%
912 \affixline@num%
913 \fi%
914 \xifinlist{\the\l@dpscL}{\eled@sections@@}%
915 {\add@inserts\affixside@note}%
916 {\print@lineL}}%
917 \add@penaltiesL%
918 \global\advance\@donereallinesL\@ne%
919 \global\advance\@donetotallinesL\@ne%
920 \else%
921 \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
922 \global\advance\@donetotallinesL\@ne%
923 \fi}
924
925
926 %

```

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

```

927 \def\print@lineL{%
928 \affixstart@numL%
929 \l@dld@ta %space kept for backward compatibility
930 \add@inserts\affixside@note%
931 \l@dlsn@te %space kept for backward compatibility
932 {\ledllfill\hb@xt@ \Lcolwidth{%
933 \do@insidelineLhook%
934 \ifluatex%
935 \textdir\l@luatextextdir@L%

```

```

936     \fi%
937     \new@lineL%
938     \inserthangingsymbolL%
939     \l@dunhbox@line{\one@line}}\ledrfill\l@drd@ta%
940 \l@drsn@te}}
941
942 %

```

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

```

943 \def\print@eledsectionL{%
944   \addtocounter{pstartL}{-1}%
945   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{
946   \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}
947   \numdef{\temp@}{\l@dpscL-1}%
948   \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
949   \@eled@sectioningtrue%
950   \bgroup%
951     \ifluatex%
952     \textdir\l@luatextextdir@L%
953     \pardir\l@luatexpardir@L%
954     \bodydir\l@luatexbodydir@L%
955     \ifdefstring{\l@luatextextdir@L}{TRT}{\@RTLtrue}{}%
956     \fi%
957     \csuse{eled@sectioning@the\l@dpscL}%
958   \egroup%
959   \@eled@sectioningfalse%
960   \global\csundef{eled@sectioning@the\l@dpscL}%
961   \if@RTL%
962     \hspace{-3\paperwidth}%
963     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
964   \else%
965     \hspace{3\paperwidth}%
966     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
967   \fi%
968   \vskip\eledsection@correcting@skip%
969 }
970
971 %

```

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

```

\doinlinedlineLhook
\doinlinedlineRhook
972 \newcommand*{\dolineLhook}[1]{\gdef\do@lineLhook{#1}}%
973 \newcommand*{\dolineRhook}[1]{\gdef\do@lineRhook{#1}}%
974 \newcommand*{\doinlinedlineLhook}[1]{\gdef\do@insidelineLhook{#1}}%
975 \newcommand*{\doinlinedlineRhook}[1]{\gdef\do@insidelineRhook{#1}}%
976
977 %

```

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

```

\do@lineRhook
\do@insidelineLhook
\do@insidelineRhook
978 \newcommand*\do@lineLhook{}
979 \newcommand*\do@lineRhook{}
980 \newcommand*\do@insidelineLhook{}
981 \newcommand*\do@insidelineRhook{}
982
983 %

```

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

```

984 \newcommand*\do@lineR{%
985   \let\linenumrep\linenumrepR%
986   \let\sublinenumrep\sublinenumrepR%
987   \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscr}%
988   \ledRcol@true%
989   \advance\countRline \@ne%
990   \ifvbox\namebox{l@dRcolrawbox\the\l@dpscr}%
991   {\vbadness=10000%
992    \splittopskip=\z@%
993    \do@lineRhook%
994    \l@emptyd@ta%
995    \global\setbox\one@lineR=\vsplit\namebox{l@dRcolrawbox\the\l@dpscr}%
996     to\baselineskip}%
997   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startR
998 }{}%
999   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
1000   \getline@numR%
1001   \ifnum\@lockR>\@ne%
1002     \inserthangingsymbolRtrue%
1003   \else%
1004     \inserthangingsymbolRfalse%
1005   \fi%
1006   \setbox\l@dtrightbox%
1007   \hb@xt@ \Rcolwidth{%
1008     \ifl@dhidenumber%
1009       \global\l@dhidenumberfalse%
1010       \f@x@l@cksR%
1011     \else%
1012       \affixline@numR%
1013     \fi%
1014     \xifinlist{\the\l@dpscr}{\eled@sectionsR@@}%
1015     {\add@insertsR\affixside@noteR}%
1016     {\print@lineR}%
1017   }%
1018   \add@penaltiesR%
1019   \global\advance\@donereallinesR\@ne%
1020   \global\advance\@donetotallinesR\@ne%
1021 \else%
1022   \setbox\l@dtrightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}%

```

```

1022 \global\advance\@donetotallinesR\@ne%
1023 \fi%
1024 \ledRcol@false%
1025 }
1026
1027
1028 %

```

```

\print@lineR
\print@eledsectionR

```

### VIII.3 Line and page number computation

`\getline@numR` The `\getline@numR` macro determines the page and line numbers for the right text line we are about to send to the vertical list. The `\getline@numL` is the same for left text.

```

1029 \newcommand*\getline@numR}{%
1030 \global\advance\absline@numR \@ne
1031 \do@actionsR
1032 \do@ballastR
1033 \ifledgroupnotesR@else
1034 \ifnumberline
1035 \ifsublines@
1036 \ifnum\sub@lockR<\tw@
1037 \global\advance\subline@numR \@ne
1038 \fi
1039 \else
1040 \ifnum\@lockR<\tw@
1041 \global\advance\line@numR \@ne
1042 \global\subline@numR \z@
1043 \fi
1044 \fi
1045 \fi
1046 \fi
1047 }
1048 \newcommand*\getline@numL}{%
1049 \global\advance\absline@num \@ne
1050 \do@actions
1051 \do@ballast
1052 \ifledgroupnotesL@else
1053 \ifnumberline
1054 \ifsublines@
1055 \ifnum\sub@lock<\tw@
1056 \global\advance\subline@num \@ne
1057 \fi
1058 \else
1059 \ifnum\@lock<\tw@
1060 \global\advance\line@num \@ne
1061 \global\subline@num \z@
1062 \fi
1063 \fi

```

```

1064     \fi
1065 \fi
1066 }
1067
1068
1069 %

```

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

```

1070 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1071 \begingroup
1072   \advance\absline@numR \@ne
1073   \ifnum\next@actionlineR=\absline@numR
1074     \ifnum\next@actionR>-1001
1075       \global\advance\ballast@count by -\c@ballast
1076     \fi
1077   \fi
1078 \endgroup}
1079 %

```

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

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

```

1080
1081 \newif\ifl@dskipversenumberR
1082 \newcommand*{\do@actions@fixedcodeR}{%
1083   \ifcase\@l@dttempcnta%
1084     \or% % 1001
1085       \global\sublines@true
1086     \or% % 1002
1087       \global\sublines@false
1088     \or% % 1003
1089       \global\@lockR=\@ne
1090     \or% % 1004%
1091       \ifnum\@lockR=\tw@
1092         \global\@lockR=\thr@@
1093       \else
1094         \global\@lockR=\z@
1095       \fi
1096     \or% % 1005
1097       \global\sub@lockR=\@ne
1098     \or% % 1006
1099       \ifnum\sub@lockR=\tw@
1100         \global\sub@lockR=\thr@@
1101       \else
1102         \global\sub@lockR=\z@
1103       \fi

```

```

1104 \or% % 1007
1105 \l@dskipnumbertrue
1106 \or% % 1008
1107 \l@dskipversenumberRtrue%
1108 \or% % 1009
1109 \l@dhidnumbertrue%
1110 \else%
1111 \led@warn@BadAction
1112 \fi%
1113 }
1114
1115
1116 \newcommand*{\do@actionsR}{%
1117 \global\let\do@actions@nextR=\relax
1118 \l@l@tempcntb=\absline@numR
1119 \ifnum\l@l@tempcntb<\next@actionlineR\else
1120 \ifnum\next@actionR>-1001\relax
1121 \global\page@numR=\next@actionR
1122 \ifbypage@R
1123 \global\line@numR \z@ \global\subline@numR \z@
1124 \fi
1125 \else
1126 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1127 \l@l@tempcnta=-\next@actionR
1128 \advance\l@l@tempcnta by -5001\relax
1129 \ifsublines@
1130 \global\subline@numR=\l@l@tempcnta
1131 \else
1132 \global\line@numR=\l@l@tempcnta
1133 \fi
1134 \else
1135 \l@l@tempcnta=-\next@actionR
1136 \advance\l@l@tempcnta by -1000\relax
1137 \do@actions@fixedcodeR
1138 \fi
1139 \fi
1140 \ifx\actionlines@listR\empty
1141 \gdef\next@actionlineR{1000000}%
1142 \else
1143 \glp\actionlines@listR\to\next@actionlineR
1144 \glp\actions@listR\to\next@actionR
1145 \global\let\do@actions@nextR=\do@actionsR
1146 \fi
1147 \fi
1148 \do@actions@nextR}
1149
1150 %

```

### VIII.4 Line number printing

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

```

1151 \newcommand*{\l@dcalcnm}[3]{%
1152 \ifnum #1 > #2\relax
1153   \l@dtempcnta = #1\relax
1154   \advance\l@dtempcnta by -#2\relax
1155   \divide\l@dtempcnta by #3\relax
1156   \multiply\l@dtempcnta by #3\relax
1157   \advance\l@dtempcnta by #2\relax
1158 \else
1159   \l@dtempcnta=#2\relax
1160 \fi}
1161
1162 \newcommand*{\ch@cksub@l@ckR}{%
1163 \ifcase\sub@lockR
1164 \or
1165   \ifnum\sublock@disp=\@ne
1166     \l@dtempcntb \z@ \l@dtempcnta \@ne
1167   \fi
1168 \or
1169   \ifnum\sublock@disp=\tw@
1170   \else
1171     \l@dtempcntb \z@ \l@dtempcnta \@ne
1172   \fi
1173 \or
1174   \ifnum\sublock@disp=\z@
1175     \l@dtempcntb \z@ \l@dtempcnta \@ne
1176   \fi
1177 \fi}
1178
1179 \newcommand*{\ch@ck@l@ckR}{%
1180 \ifcase\@lockR
1181 \or
1182   \ifnum\lock@disp=\@ne
1183     \l@dtempcntb \z@ \l@dtempcnta \@ne
1184   \fi
1185 \or
1186   \ifnum\lock@disp=\tw@
1187   \else
1188     \l@dtempcntb \z@ \l@dtempcnta \@ne
1189   \fi
1190 \or
1191   \ifnum\lock@disp=\z@
1192     \l@dtempcntb \z@ \l@dtempcnta \@ne
1193   \fi
1194 \fi}
1195
1196

```

```

1197 \newcommand*{\f@x@l@cksR}{%
1198   \ifcase\@lockR
1199   \or
1200     \global\@lockR \tw@
1201   \or \or
1202     \global\@lockR \z@
1203   \fi
1204   \ifcase\sub@lockR
1205   \or
1206     \global\sub@lockR \tw@
1207   \or \or
1208     \global\sub@lockR \z@
1209   \fi}
1210
1211
1212 \newcommand*{\affixline@numR}{%
1213   \ifledgroupnotesR\else\ifnumberline
1214   \ifl@dskipnumber
1215     \global\l@dskipnumberfalse
1216   \else
1217     \ifsublines@
1218       \@l@tempcntb=\subline@numR
1219       \l@dcalcnm{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR
1220     }%
1221     \ch@cksub@lockR
1222   \else
1223     \@l@tempcntb=\line@numR
1224     \ifx\linenumberlist\empty
1225       \l@dcalcnm{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1226     \else
1227       \@l@tempcnta=\line@numR
1228       \edef\rem@inder{\linenumberlist,\number\line@numR,}%
1229       \edef\sc@n@list{\def\noexpand\sc@n@list
1230         ###1,\number\@l@tempcnta,###2|\def\noexpand\rem@inder{###2}}%
1231       \sc@n@list\expandafter\sc@n@list\rem@inder|
1232       \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1233     \fi
1234     \ch@ck@l@ckR
1235   \fi
1236   \ifnum\@l@tempcnta=\@l@tempcntb
1237   \ifl@dskipversenumberR\else
1238     \if@twocolumn
1239     \if@firstcolumn
1240       \gdef\l@dld@ta{\llap{\leftlinenumR}}%
1241     \else
1242       \gdef\l@drd@ta{\rlap{\rightlinenumR}}%
1243     \fi
1244   \else
1245     \@l@tempcntb=\line@marginR
1246     \ifnum\@l@tempcntb>\@ne

```



```

1246     \advance\@l@tempcntb by\page@numR
1247     \fi
1248     \ifodd\@l@tempcntb
1249     \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1250     \else
1251     \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1252     \fi
1253   \fi
1254 \fi
1255 \fi
1256 \f@x@l@cksR
1257 \fi
1258 \fi
1259 \fi}
1260 %

```

## VIII.5 Pstart number printing in side

The printing of the pstart number is like in `reledmac`, 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@numL1261
\affixpstart@numR1262 \newcommand*{\affixpstart@numL}{%
  \leftpstartnum1263 \ifsidepstartnum
  \rightpstartnumR1264 \if@twocolumn
  \leftpstartnumL1265 \if@firstcolumn
  \rightpstartnumL1266 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
  \ifpstartnumR1267 \else
1268     \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1269     \fi
1270   \else
1271     \@l@tempcntb=\line@margin
1272     \ifnum\@l@tempcntb>\@ne
1273       \advance\@l@tempcntb \page@num
1274     \fi
1275     \ifodd\@l@tempcntb
1276     \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1277     \else
1278     \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1279     \fi
1280   \fi
1281 \fi
1282 }
1283 \newcommand*{\affixpstart@numR}{%

```

```

1284 \ifsidepstartnum
1285 \if@twocolumn
1286   \if@firstcolumn
1287     \gdef\l@dld@ta{\llap{{\leftpstartnumR}}}%
1288   \else
1289     \gdef\l@dld@ta{\rlap{{\rightpstartnumR}}}%
1290   \fi
1291 \else
1292   \l@dttempcntb=\line@marginR
1293   \ifnum\l@dttempcntb>\@ne
1294     \advance\l@dttempcntb \page@numR
1295   \fi
1296   \ifodd\l@dttempcntb
1297     \gdef\l@dld@ta{\rlap{{\rightpstartnumR}}}%
1298   \else
1299     \gdef\l@dld@ta{\llap{{\leftpstartnumR}}}%
1300   \fi
1301 \fi
1302 \fi
1303 }
1304
1305 \newcommand*{\leftpstartnumL}{
1306 \ifpstartnum
1307 \thepstartL
1308 \kern\linenumsep\global\pstartnumfalse\fi
1309 }
1310 \newcommand*{\rightpstartnumL}{
1311 \ifpstartnum\kern\linenumsep
1312 \thepstartL
1313 \global\pstartnumfalse\fi
1314 }
1315 \newif\ifpstartnumR
1316 \pstartnumRtrue
1317 \newcommand*{\leftpstartnumR}{
1318 \ifpstartnumR
1319 \thepstartR
1320 \kern\linenumsep\global\pstartnumRfalse\fi
1321 }
1322 \newcommand*{\rightpstartnumR}{
1323 \ifpstartnumR\kern\linenumsep
1324 \thepstartR
1325 \global\pstartnumRfalse\fi
1326 }
1327 %

```

### VIII.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.

```

1328 \list@create{\inserts@listR}
1329 %

\add@insertsR The right text version.
\add@inserts@nextR
1330 \newcommand*{\add@insertsR}{%
1331 \global\let\add@inserts@nextR=\relax
1332 \ifx\inserts@listR\empty \else
1333 \ifx\next@insertR\empty
1334 \ifx\insertlines@listR\empty
1335 \global\noteschanged@true
1336 \gdef\next@insertR{100000}%
1337 \else
1338 \gl@p\insertlines@listR\to\next@insertR
1339 \fi
1340 \fi
1341 \ifnum\next@insertR=\absline@numR
1342 \gl@p\inserts@listR\to\@insertR
1343 \@insertR
1344 \global\let\@insertR=\undefined
1345 \global\let\next@insertR=\empty
1346 \global\let\add@inserts@nextR=\add@insertsR
1347 \fi
1348 \fi
1349 \add@inserts@nextR}
1350
1351 %

```

## VIII.7 Penalties

`\add@penaltiesL` `\add@penaltiesR` `\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 are 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 does not go below  $-10000$ .

```

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

```

```

\ifnum\@l@dttempcntb=\num@linesR
  \advance\@l@dttempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
  \advance\@l@dttempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@dttempcnta=\z@
  \relax
\else
  \ifnum\@l@dttempcnta>-10000
    \penalty\@l@dttempcnta
  \else
    \penalty -10000
  \fi
\fi}

```

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

```

1352 \newcommand*\add@penaltiesL}{
1353 \newcommand*\add@penaltiesR}{
1354
1355 %

```

### VIII.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.

```

1356 \newcommand*\flush@notesR}{%
1357   \@xloop
1358   \ifx\inserts@listR\empty \else
1359     \glp\inserts@listR\to\@insertR
1360     \@insertR
1361     \global\let\@insertR=\undefined
1362   \repeat}
1363
1364 %

```

## IX Footnotes

### IX.1 Line number printing

The `\printlinesR` 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 V.9 p. 81 of Eledmac’ handbook: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

`\printlinesR` This is the right text version of `\printlines` and takes account of `\@Rlineflag`. Just a reminder of the arguments:

```
\printlinesR #1 | #2 | #3 | #4 | #5 | #6 | #7
\printlinesR start-page | line | subline | end-page | line | subline | font
```

```
1365 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1366 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1367 \ifl@d@pnum #1\fullstop\fi
1368 \ifledplinenum \linenumr@p{#2}\@Rlineflag\else \symplinenum\fi
1369 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1370 \ifl@d@dash \endashchar\fi
1371 \ifl@d@pnum #4\fullstop\fi
1372 \ifl@d@elin \linenumr@p{#5}\@Rlineflag\fi
1373 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1374 \endgroup}
1375
1376
1377 %
```

### IX.2 Footnotes output specific to `\Pages`

`\print@Xnotes@forpages` `\Xonlyside` and `\onlysideX` hooks for `\Pages` allow notes to be printed either in left or right pages only. The implementation of such features is delegated to `\correct@Xfootins@box` `\print@notesX@forpages` `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed<sup>3</sup>:

- If notes are to be printed in both sides, we just proceed the usual way: print the foot starts for the series, then the foot group.
- If notes are to be printed in the left side, we do these prints only for even pages ; if notes are to be printed in the right side, we do these prints only for odd pages.
- However, that is not enough. Because the problem does not only consists in printing notes in any particular page. It is also not to put aside room for notes in the pages where we do not want to print them. To take an example: if some note in the left side is too long by 160pt to be printed in full in the left page, we do not want to put aside 160pt a space for it in the following right page.

<sup>3</sup>See <http://tex.stackexchange.com/a/230332/7712>.

- To solve this problem, we change the magnification factor associated with notes before going to the next page. If we start a page where no notes are supposed to be printed, the magnification counter is set to 0. We also set the note skip to 0pt. Before starting a new page where these notes are supposed to be printed, we reset these counter and skip to their default values. (About these counter and skip, read *The TeXbook* p. 122-125).
- There still remains a last problem. This problem is quite complex to understand, so an example will speak for itself. Suppose we allow 10 lines of notes by page. Suppose a long note, be it 25 lines, which needs three pages to be printed. Suppose it must be printed only on left pages, namely odd pages.

On p. 2, the first 10 lines of the notes are printed. On p. 3, the box associated to the notes contains 10 lines. However, as we are in a right page, we do not void this box. So  $\TeX$  will keep its content for the pages to come. However, on p. 4 it will also add one line in the footnote box, because in any case,  $\TeX$  adds some content in the box when preparing the output routines, even if there is some content left in this box from the previous pages. So the lines in the note box at p. 4 will be  $10 + 1 = 11$ . There is one line which should not be there. Furthermore, as the box size is for 10 lines and not for 11 lines, this last line will be glued to the previous one.

To fix this double issue:

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

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

```
1378 \newcommand\print@Xnotes@forpages[1]{%
1379 %
```

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

```
1380 \ifcsemtty{Xonlyside@#1}{%
1381 \csuse{#1footstart}{#1}%
1382 \csuse{#1footgroup}{#1}%
1383 }%
1384 %
```

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

```

1385   {%
1386     \ifboolexpr{%
1387       ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\ifnumodd{\c@page
1388       }})%
1389       or%
1390       (test {\ifcsstring{Xonlyside@#1}{R}} and test{\ifnumodd{\c@page}})%
1391     }%

```

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

```

1392   {%
1393     \correct@Xfootins@box{#1}%
1394     \csuse{#1footstart}{#1}%
1395     \csuse{#1footgroup}{#1}%
1396   }%

```

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

```

1397     \global\count\csuse{#1footins}=0%
1398     \global\skip\csuse{#1footins}=0pt%
1399   }%

```

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

```

1400     \csuse{Xnotefontsize@#1}%
1401     \global\advance\dimen\csuse{#1footins} by -\baselineskip%
1402   }%

```

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

```

1403     \global\boolfalse{keepforXside@#1}%
1404   }%
1405 }%

```

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

```

1406   {%
1407     \global\booltrue{keepforXside@#1}%
1408   }%

```

Then restore expected rooms for notes on the next page.

```

1409     \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1410     \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
1411   }%

```

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

```

1412     \bgroup%
1413         \csuse{Xnotefontsize@#1}%
1414         \global\advance\dimen\csuse{#1footins} by \baselineskip%
1415     \egroup%
1416 %

```

```

1417 % End of \protect\cs{print@Xnotes@forpages}.
1418     }%
1419 }%
1420 }%
1421 %

```

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

```

1422 \newcommand{\correct@Xfootins@box}[1]{%
1423 %

```

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

```

1424     \ifbool{keepforXside@#1}{%
1425 %

```

Some setting needed to do the right splitting.

```

1426         \csuse{Xnotefontsize@#1}%
1427         \splittopskip=0pt%
1428 %

```

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

```

1429     \global\setbox\csuse{#1footins}=\vbox{%
1430         \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1431         \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1432     \unvbox\csuse{#1footins}%
1433     }%
1434 %

```

End of the macro.

```

1435     }{}%
1436 }%
1437 %

```

And now, the same for familiar footnotes.

```

1438 \newcommand\print@notesX@forpages[1]{%
1439     \ifcsemtyp{onlysideX@#1}{%
1440         \csuse{footstart#1}{#1}%
1441         \csuse{footgroup#1}{#1}%
1442     }%
1443     {%
1444         \ifboolexpr{%

```



```

1445 ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page
}})%
1446 or%
1447 (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}})%
1448 }%
1449 {%
1450 \correct@footinsX@box{#1}%
1451 \csuse{footstart#1}{#1}%
1452 \csuse{footgroup#1}{#1}%
1453 \global\count\csuse{footins#1}=0%
1454 \global\skip\csuse{footins#1}=0pt%
1455 \csuse{notefontsizeX@#1}%
1456 \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1457 \global\boolfalse{keepforsideX@#1}%
1458 }%
1459 {%
1460 \global\booltrue{keepforsideX@#1}%
1461 \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1462 \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1463 \bgroup%
1464 \csuse{notefontsizeX@#1}%
1465 \global\advance\dimen\csuse{footins#1} by \baselineskip%
1466 \egroup%
1467 }%
1468 }%
1469 }%
1470 \newcommand{\correct@footinsX@box}[1]{%
1471 \ifbool{keepforsideX@#1}{%
1472 \csuse{notefontsizeX@#1}%
1473 \splittopskip=0pt%
1474 \global\setbox\csuse{footins#1}=\vbox{%
1475 \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1476 \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1477 \unvbox\csuse{footins#1}%
1478 }%
1479 }{}%
1480 }%
1481 %

```

## X 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.

```

1482 \list@create{\labelref@listR}
1483
1484 %

```

`\edlabel` This command is defined only one time in `reledmac`, including features for `reledpar`.

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

```

1485 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1486   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1487     \led@warn@DuplicateLabel{#4}%
1488   \fi
1489   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\@Rlineflag|#3|#4}%
1490   \ignorespaces}
1491 \AtBeginDocument{%
1492   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1493 }
1494 %
1495 %

```

`\@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.

It is defined on `reledmac`.

## XI Side notes

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

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

`\sidenotemargin*`

```

1496 \WithSuffix\newcommand\sidenotemargin*[1]{%
1497   \l@dgetsidenote@margin{#1}
1498   \global\sidenote@marginR=\@l@tempcntb
1499   \global\sidenote@margin=\@l@tempcntb
1500 }
1501 \newcount\sidenote@marginR
1502 \global\sidenote@margin=\@ne
1503 %
1504 %

```

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

```

1505 \newcommand*{\affixside@noteR}{%
1506   \def\sidenotecontent@{}%
1507   \numgdef{\itemcount@}{0}%
1508   \def\do##1{%
1509     \ifnumequal{\itemcount@}{0}%
1510     {%

```

```

1511     \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
1512     {\appto\sidenotecontent@\sidenotesep ##1}%
1513     }%
1514     \numgdef{\itemcount@}{\itemcount@+1}%
1515   }%
1516   \dolistloop{\l@dcsnotetext}%
1517   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1518   \gdef\@templ@d{}%
1519   \gdef\@templ@n{\l@dcsnotetext\l@dcsnotetext@l\l@dcsnotetext@r}%
1520   \ifx\@templ@d\@templ@n \else%
1521     \iftwocolumn%
1522       \iffirstcolumn%
1523         \setl@dlp@rbox{##1}{\sidenotecontent@}%
1524       \else%
1525         \setl@drp@rbox{\sidenotecontent@}%
1526       \fi%
1527     \else%
1528       \@l@dttempcntb=\sidenote@marginR%
1529       \ifnum\@l@dttempcntb>\@ne%
1530         \advance\@l@dttempcntb by\page@numR%
1531       \fi%
1532       \ifodd\@l@dttempcntb%
1533         \setl@drp@rbox{\sidenotecontent@}%
1534         \gdef\sidenotecontent@{}%
1535         \numdef{\itemcount@}{0}%
1536         \dolistloop{\l@dcsnotetext@l}%
1537         \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1538         \setl@dlp@rbox{\sidenotecontent@}%
1539       \else%
1540         \setl@dlp@rbox{\sidenotecontent@}%
1541         \gdef\sidenotecontent@{}%
1542         \numdef{\itemcount@}{0}%
1543         \dolistloop{\l@dcsnotetext@r}%
1544         \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1545         \setl@drp@rbox{\sidenotecontent@}%
1546       \fi%
1547     \fi%
1548   \fi%
1549 }
1550 %
1551 %

```

## XII Familiar footnotes

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original `\@footnotetext`. There are both defined in `reledmac`.

`\normalbfnoteX`

### XIII Verse

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

```

\inserthangingsymbolL552 \newif\ifinserthangingsymbolR
\inserthangingsymbolR553 \newcommand{\inserthangingsymbolL}{%
1554   \ifinserthangingsymbol%
1555     \ifinstanzaL%
1556       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1557         \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1558       \@hangingsymbol%
1559     \fi%
1560   \fi%
1561 }%
1562 \newcommand{\inserthangingsymbolR}{%
1563   \ifinserthangingsymbolR%
1564     \ifinstanzaR%
1565       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1566         \noexpand\csname sza@0@\endcsname}\stanzaindentbase%
1567       \@hangingsymbol%
1568     \fi%
1569   \fi%
1570 }%
1571 %

```

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.

```

1572   \chardef\next=\catcode`\&
1573   \catcode`\&=\active
1574
1575 %

```

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

```

1576 \newenvironment{astanza}[1][ ]{%
1577   \catcode`\&\active
1578   \global\stanza@count\@ne\stanza@modulo\@ne
1579   \ifnum\usernamecount{sza@00}=\z@
1580     \let\stanza@hang\relax
1581     \let\endlock\relax
1582   \else
1583     \rightskip\z@ plus 1fil\relax
1584   \fi
1585   \ifnum\usernamecount{szp@00}=\z@
1586     \let\sza@penalty\relax

```

```

1587 \fi
1588 \def&{%
1589   \endlock\mbox{}%
1590   \sza@penalty
1591   \global\advance\stanza@count\@ne
1592   \@astanza@line}%
1593 \def\&{\@stopastanza}%
1594 \pstart[#1]%
1595 \@astanza@line
1596 }{}
1597
1598 %

```

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

```

1599 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1600   \endlock\mbox{}%
1601   \pend[#1]%
1602 }%
1603 %

```

**\@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.

```

1604 \newcommand*{\@astanza@line}{%
1605   \ifnum\value{stanzaindentsrepetition}=0
1606     \parindent=\csname sza@\number\stanza@count
1607       @\endcsname\stanzaindentbase
1608   \else
1609     \parindent=\csname sza@\number\stanza@modulo
1610       @\endcsname\stanzaindentbase
1611     \managestanza@modulo
1612   \fi
1613   \par
1614   \stanza@hang%\mbox{}%
1615   \ignorespaces}
1616
1617 %

```

Lastly reset the modified category codes.

```

1618 \catcode`\&=\next
1619
1620 %

```

**\thestanzaL** And now, the left and right stanza counter.

```

\thestanzaR
1621 \newcounter{stanzaL}
1622 \newcounter{stanzaR}
1623 \renewcommand{\thestanzaL}{%

```

```

1624 \textbf{\arabic{stanzaL}}%
1625 }
1626 \renewcommand{\thestanzaR}{%
1627 \textbf{\arabic{stanzaR}}%
1628 }
1629 %
1630 %

```

## XIV Naming macros

The  $\LaTeX$  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’.

```

\setnamebox
\unhnamebox
\unvnamebox
\namebox
1631 \providecommand*\newnamebox}[1]{%
1632 \expandafter\newbox\csname #1\endcsname}
1633 \providecommand*\setnamebox}[1]{%
1634 \expandafter\setbox\csname #1\endcsname}
1635 \providecommand*\unhnamebox}[1]{%
1636 \expandafter\unhbox\csname #1\endcsname}
1637 \providecommand*\unvnamebox}[1]{%
1638 \expandafter\unvbox\csname #1\endcsname}
1639 \providecommand*\namebox}[1]{%
1640 \csname #1\endcsname}
1641
1642 %

```

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

```

\usernamecount
1643 \providecommand*\newnamecount}[1]{%
1644 \expandafter\newcount\csname #1\endcsname}
1645 \providecommand*\usernamecount}[1]{%
1646 \csname #1\endcsname}
1647
1648 %

```

## XV Fixing babel and polyglossia

With parallel texts there is the possibility that the two sides might use different languages via `babel`. On the other hand, nor `babel` nor `polyglossia` 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
\l@dusedbabeltrue1649
1650%

```

`\l@dchecklang`

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

```

1651\patchcmd{\bbl@set@language}%
1652{\select@language{\language}}%
1653{\edef\language{#1}\select@language{\language}}%
1654{}%
1655{}%
1656
1657%

```

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

```

\selectlanguage \selectlanguage is a babel command. \theledlanguageL and \theledlanguageR
\l@duselanguage are the names of the languages of the left and right texts. \l@duselanguage is similar
\theledlanguageL to \selectlanguage.
\theledlanguageR
1658\newcommand*{\l@duselanguage}[1]{%
1659\gdef\theledlanguageL{#1}
1660\gdef\theledlanguageR{#1}
1661
1662%

```

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

```

1663\AtBeginDocument{%
1664\@ifundefined{xpg@main@language}{%
1665\@ifundefined{bbl@main@language}{%
1666%

```

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

```

1667\l@dusedbabelfalse
1668}{%
1669%

```

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.

```

1670 \l@dusedbabeltrue
1671 \let\l@doldselectlanguage\selectlanguage
1672 \let\l@doldbbl@set@language\bbl@set@language
1673 \renewcommand{\selectlanguage}[1]{%
1674   \l@doldselectlanguage{#1}%
1675   \ifledRcol \gdef\theledlanguageR{#1}%
1676   \else      \gdef\theledlanguageL{#1}%
1677   \fi}
1678 %

```

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

```

1679 \renewcommand*{\l@duselanguage}[1]{%
1680   \l@doldselectlanguage{#1}}
1681 %

```

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

```

1682 \gdef\theledlanguageL{\bbl@main@language}%
1683 \gdef\theledlanguageR{\bbl@main@language}%
1684 }%
1685 }
1686 %

```

If use polyglossia

```

1687 { \let\old@otherlanguage\otherlanguage%
1688   \renewcommand{\otherlanguage}[2][ ]{%
1689     \selectlanguage{#1}{#2}%
1690     \ifledRcol \gdef\theledlanguageR{#2}%
1691     \else      \gdef\theledlanguageL{#2}%
1692     \fi}%
1693   \let\l@duselanguage\select@language%
1694   \gdef\theledlanguageL{\xpg@main@language}%
1695   \gdef\theledlanguageR{\xpg@main@language}%
1696 %

```

That is it.

```

1697 }}
1698 %

```

## XVI 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 default is  
`\l@dc@maxchunks` 5120 chunk pairs.

```
1699 \newcount\l@dc@maxchunks
1700 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1701 \maxchunks{5120}
1702
1703 %
```

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

`\l@dnumpstartsR`

```
1704 \newcount\l@dnumpstartsR
1705
1706 %
```

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

`\l@pscR`

```
1707 \newcount\l@dpscl
1708 \newcount\l@dpscR
1709
1710 %
```

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

```
1711 \newcommand*{\l@dsetuprawboxes}{%
1712 \l@dtempcntb=\l@dc@maxchunks
1713 \loop\ifnum\l@dtempcntb>\z@
1714 \newnamebox{\l@dLcolrawbox\the\l@dtempcntb}
1715 \newnamebox{\l@dRcolrawbox\the\l@dtempcntb}
1716 \advance\l@dtempcntb \m@ne
1717 \repeat}
1718
1719 %
```

`\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.

```
1720 \newcommand*{\l@dsetupmaxlinecounts}{%
1721 \l@dtempcntb=\l@dc@maxchunks
1722 \loop\ifnum\l@dtempcntb>\z@
1723 \newnamecount{\l@dmaxlinesinpar\the\l@dtempcntb}
1724 \advance\l@dtempcntb \m@ne
1725 \repeat}
1726 \newcommand*{\l@dzeromaxlinecounts}{%
1727 \begingroup
1728 \l@dtempcntb=\l@dc@maxchunks
1729 \loop\ifnum\l@dtempcntb>\z@
```

```

1730 \global\usernamecount{l@dmxlinesinpar\the\l@dttempcntb}=\z@
1731 \advance\l@dttempcntb \m@ne
1732 \repeat
1733 \endgroup}
1734
1735 %

```

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

```

1736 \AtBeginDocument{%
1737 \l@dssetuprawboxes
1738 \l@dssetupmaxlinecounts
1739 \l@dzeromaxlinecounts
1740 \l@dnumpstartsL=\z@
1741 \l@dnumpstartsR=\z@
1742 \l@dpscL=\z@
1743 \l@dpscR=\z@}
1744
1745 %

```

## XVII Checking text to be processed

```

\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue
\@pstartsfalse
\check@pstarts
1746 \newif\if@pstarts
1747 \newcommand*\check@pstarts}{%
1748 \@pstartsfalse
1749 \ifnum\l@dnumpstartsL>\l@dpscL
1750 \@pstartstrue
1751 \else
1752 \ifnum\l@dnumpstartsR>\l@dpscR
1753 \@pstartstrue
1754 \fi
1755 \fi
1756 }
1757
1758 %

```

```

\if@raw@text \check@raw@text checks whether the current Left or Right box is void or not. If
\@raw@texttrue one or other is not void it sets \@raw@texttrue, otherwise both are void and it sets
\@raw@textfalse \@raw@textfalse.
\check@raw@text
1759 \newif\if@raw@text
1760 \newcommand*\check@raw@text}{%
1761 \@raw@textfalse
1762 \ifvbox\namebox{l@dLcolrawbox\the\l@dpscL}
1763 \@raw@texttrue
1764 \else

```

```

1765 \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}
1766 \araw@texttrue
1767 \fi
1768 \fi
1769 }
1770 %
1771 %

```

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

```

1772 \newcommand*\@writelinesinparL{%
1773 \edef\next{%
1774 \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1775 \next
1776 \global\@donereallinesL \z@}
1777 \newcommand*\@writelinesinparR{%
1778 \edef\next{%
1779 \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1780 \next
1781 \global\@donereallinesR \z@}
1782 %
1783 %

```

## XVIII Parallel columns

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

```

1784 \newsavebox{\@eledsectionL}%
1785 \newsavebox{\@eledsectionR}%
1786 %

```

`\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.

```

1787 \newcommand*\Columns{%
1788 \ifl@dpairing%
1789 \led@err@Columns@InsideEnv%
1790 \fi%
1791 \l@dprintingcolumnstrue%
1792 \eledsection@correcting@skip=-\baselineskip% Correction for sections'
titles
1793 \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1794 \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1795 \fi
1796 %

```

Start a group and zero counters, etc.

```

1797 \begingroup
1798 \l@dzeropenalties
1799 \endgraf\global\num@lines=\prevgraf
1800 \global\num@linesR=\prevgraf
1801 \global\par@line=\z@
1802 \global\par@lineR=\z@
1803 \global\l@dpscL=\z@
1804 \global\l@dpscR=\z@
1805 %

```

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

```

1806 \check@pstarts
1807 \loop\if@pstarts
1808 \global\pstartnumtrue
1809 \global\pstartnumRtrue
1810 %

```

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

```

1811 \global\advance\l@dpscL \@ne
1812 \global\advance\l@dpscR \@ne
1813 \restore@pstartL@pc%
1814 \restore@pstartR@pc%
1815 %

```

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

```

1816 \Columns@print@before@pstart%
1817 %

```

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

```

1818 \checkraw@text
1819 { \loop\ifaraw@text
1820 %

```

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

```

1821 \l@duselanguage{\theledlanguageL}%
1822 \do@lineL
1823 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1824 {%
1825 \ifdefstring{\@eledsectmark}{L}%
1826 {\csuse{eled@sectmark@\the\l@dpscL}%
1827 }}%
1828 \global\csundef{eled@sectmark@\the\l@dpscL}%
1829 \savebox{\@eledsectionL}{\parbox[t][t]{\Lcolwidth}{\vbox
}\print@eledsectionL}}%\vbox{}-> prevent alignment troubles with RTL
language

```

```

1830     }%
1831     }%
1832     \l@duselanguage{\theledlanguageR}%
1833     \do@lineR
1834     \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1835     {%
1836     \ifdefstring{\@eledsectmark}{R}%
1837     {\csuse{eled@sectmark@\the\l@dpscR R}%
1838     }%
1839     \global\csundef{eled@sectmark@\the\l@dpscR R}%
1840     \savebox{\@eledsectionR}{\parbox[t][t]{\Rcolwidth}{\vbox
\print@eledsectionR}}% \vbox{}-> prevent alignment troubles with RTL
language
1841     }%
1842     \hb@xt@ \hsize{%
1843     \ifdefstring{\columns@position}{L}{\hfill }%
1844     \unhbox\l@dleftbox%
1845     \ifhbox\@eledsectionL%
1846     \usebox{\@eledsectionL}%
1847     \fi%
1848     \print@columnseparator%
1849     \unhbox\l@drightbox%
1850     \ifhbox\@eledsectionR%
1851     \usebox{\@eledsectionR}%
1852     \fi%
1853     \ifdefstring{\columns@position}{R}{\hfill}%
1854     }%
1855     \checkraw@text
1856     \checkverseL
1857     \checkverseR
1858     \checkpb@columns
1859     \repeat}
1860 %

```

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 is by pstart.

```

1861     \@writelinesinparL
1862     \@writelinesinparR
1863     \check@pstarts
1864     \ifbypstart@%
1865     \write\linenum@out{\string\@set[1]}
1866     \resetprevline@
1867     \fi
1868     \ifbypstart@R
1869     \write\linenum@outR{\string\@set[1]}
1870     \resetprevline@
1871     \fi
1872     \Columns@print@after@pend%
1873     \repeat

```

```

1874 %
    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.
1875     \flush@notes
1876     \flush@notesR
1877 \endgroup
1878 %
1879 \global\l@dpscL=\z@
1880 \global\l@dpscR=\z@
1881 \global\l@dnumstartsL=\z@
1882 \global\l@dnumstartsR=\z@
1883 \l@dprintingcolumnsfalse%
1884 \ignorespaces
1885 \global\instanzaLfalse
1886 \global\instanzaRfalse}
1887
1888 %

```

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

```

1889 \def\print@columnseparator{%
1890   \ifdim\beforecolumnseparator<0pt%
1891     \hfill%
1892   \else%
1893     \hspace{\beforecolumnseparator}%
1894   \fi%
1895   \columnseparator%
1896   \ifdim\aftercolumnseparator<0pt%
1897     \hfill%
1898   \else%
1899     \hspace{\beforecolumnseparator}%
1900   \fi%
1901 }%
1902 %

```

`\checkpb@columns` `\checkpb@columns` prevent or make pagebreaking in columns, depending of the use of `\ledpb` or `\lednopb`.

```

1903
1904 \newcommand{\checkpb@columns}{%
1905   \newif\if@pb
1906   \newif\if@nopb
1907   \IfStrEq{\led@pb@setting}{before}{
1908     \numdef{\next@absline}{\the\absline@num+1}%
1909     \numdef{\next@abslineR}{\the\absline@numR+1}%

```

```

1910 \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
1911 \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
1912 \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
1913 \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
1914 }{}
1915 \IfStrEq{\led@pb@setting}{after}{
1916 \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
1917 \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
1918 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
1919 \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
1920 }{}
1921 \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
1922 \if@pb\pagebreak[4]\fi
1923 }
1924 %

```

**\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).

```

1925 \newcommand*{\columnseparator}{%
1926 \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}
1927 \newdimen\columnrulewidth
1928 \columnrulewidth=\z@
1929
1930 %

```

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

```

1931 \newcommand*{\columnspan}[1]{%
1932 \xdef\columns@position{#1}%
1933 }%
1934 \xdef\columns@position{R}%
1935 %

```

**\beforecolumnseparator** `\beforecolumnseparator` and `\aftercolumnseparator` lengths are defined to -1pt.

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

```

1936 \newlength{\beforecolumnseparator}%
1937 \setlength{\beforecolumnseparator}{-2pt}%
1938
1939 \newlength{\aftercolumnseparator}%
1940 \setlength{\aftercolumnseparator}{-2pt}%
1941
1942 %

```

```

setwidthliketwocolumns@L The \setwidth... macros are called in \beginnumbering in a non-parallel typesetting
setpositionliketwocolumns@L context, to fix the width of the lines to be vertically aligned with parallel columns. They
setnotepositionliketwocolumns@L are also called at the beginning of a note's group, if some options are enabled. The
setwidthliketwocolumns@C \setposition... macros are called in \beginnumbering in a non-parallel typesetting
setpositionliketwocolumns@C context to fix the position of the lines. The \setnoteposition... macros are called in
setnotepositionliketwocolumns@C \xxxfootstart in a non-parallel typesetting context to fix the position of notes block.
setwidthliketwocolumns@R
setpositionliketwocolumns@R
setnotepositionliketwocolumns@R
1943 \newcommand{\setwidthliketwocolumns@L}{%
1944 % Temporary dimension, initially equal to the standard hsize, i.e. text
width
1945 % \begin{macrocode}
1946 \newdimen\temp%
1947 \temp=\hsize%
1948 %

Hsize : Left + Right width

1949 \hsize=\Lcolwidth%
1950 \advance\hsize\Rcolwidth%
1951 %

Now, calculating the remaining space

1952 \advance\temp-\hsize%
1953 %

And multiply the hsize by 2/3 of this space

1954 \multiply\temp by 2%
1955 \divide\temp by 3%
1956 \advance\hsize\temp%
1957 }%
1958
1959 \newcommand{\setpositionliketwocolumns@L}{%
1960 \renewcommand{\ledrlfill}{\hfill}%
1961 }%
1962
1963 \newcommand{\setnotespositionliketwocolumns@L}{%
1964 }%
1965
1966
1967 %

1968 \newcommand{\setwidthliketwocolumns@C}{%
1969 % Temporary dimension, initially equal to the standard hsize, i.e. text
width
1970 %

1971 \newdimen\temp%
1972 \temp=\hsize%
1973 % Hsize : Left + Right width
1974 %

```



```

1975 \hsize=\Lcolwidth%
1976 \advance\hsize\Rcolwidth%
1977 % Now, calculating the remaining space
1978 %

```

```

1979 \advance\temp-\hsize%
1980 %

```

And multiply the hsize by 1/2 of this space

```

1981 \divide\temp by 2%
1982 \advance\hsize\temp%
1983 }%
1984
1985 \newcommand{\setpositionliketwocolumns@C}{%
1986 \doinsidelinehook{\hfill}%
1987 \renewcommand{\ledrlfill}{\hfill}%
1988 }%
1989
1990 \newcommand{\setnotespositionliketwocolumns@C}{%
1991 \newdimen\temp%
1992 \newdimen\tempa%
1993 \temp=\hsize%
1994 \tempa=\Lcolwidth%
1995 \advance\tempa\Rcolwidth%
1996 \advance\temp-\tempa%
1997 \divide\temp by 2%
1998 \leftskip=\temp%
1999 \rightskip=-\temp%
2000 }%
2001
2002 \newcommand{\setwidthliketwocolumns@R}{%
2003 %

```

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

```

2004 \newdimen\temp%
2005 \temp=\hsize%
2006 %

```

Hsize : Left + Right width

```

2007 \hsize=\Lcolwidth%
2008 \advance\hsize\Rcolwidth%
2009 %

```

Now, calculating the remaining space

```

2010 \advance\temp-\hsize%
2011 %

```

And multiply the hsize by 2/3 of this space

```

2012 \multiply\temp by 2%
2013 \divide\temp by 3%
2014 \advance\hsize\temp%
2015 }%
2016
2017 \newcommand{\setpositionliketwocolumns@R}{%
2018 \doinsidelinehook{\hfill}%
2019 }%
2020
2021 \newcommand{\setnotespositionliketwocolumns@R}{%
2022 \newdimen\temp%
2023 \newdimen\tempa%
2024 \temp=\hsize%
2025 \tempa=Lcolwidth%
2026 \advance\tempa\Rcolwidth%
2027 \advance\temp-\tempa%
2028 \divide\temp by 2%
2029 \leftskip=\temp%
2030 \rightskip=-\temp%
2031 }%
2032
2033 %

```

`\Columns@print@before@pstart` and `\Columns@print@after@pend` print the content of the optional argument of `\pstart` / `\pend`. If this content is not empty, it also print the separator.

```

2034 \newcommand{\Columns@print@before@pstart}{%
2035 \ifboolexpr{%
2036 test{\ifcsstring{before@pstartL@the\l@dpscL}{\at@every@pstart}}%
2037 and test {\ifcsstring{before@pstartR@the\l@dpscR}{\at@every@pstart}}%
2038 and test {\ifdefempty{\at@every@pstart}}}%
2039 {}%
2040 {%
2041 \hbext@ \hsize{%
2042 \ifdefstring{\columns@position}{L}{\hfill }%
2043 \par\parbox[t] [] [t]{Lcolwidth}{%
2044 \csuse{before@pstartL@the\l@dpscL}%
2045 }%
2046 \print@columnseparator%
2047 \parbox[t] [] [t]{Rcolwidth}{%
2048 \initnumbering@sectcountR%
2049 \csuse{before@pstartR@the\l@dpscR}%
2050 }%
2051 \ifdefstring{\columns@position}{R}{\hfill}%
2052 }%
2053 }%
2054 \global\csundef{before@pstartL@the\l@dpscL}%
2055 \global\csundef{before@pstartR@the\l@dpscR}%
2056 }%

```

```

2057 \newcommand{\Columns@print@after@pend}{%
2058   \ifboolexpr{%
2059     test{\ifcsstring{after@pendL@the\l@dpscl}{\at@every@pend}}%
2060     and test {\ifcsstring{after@pendR@the\l@dpscR}{\at@every@pend}}%
2061     and test {\ifdefempty{\at@every@pend}}}%
2062     {%
2063     {%
2064       \hb@xt@ \hsize{%
2065         \ifdefstring{\columns@position}{L}{\hfill }%
2066         \parbox[t] [] [t]{\Lcolwidth}{%
2067           \csuse{after@pendL@the\l@dpscl}%
2068         }%
2069         \print@columnseparator%
2070         \parbox[t] [] [t]{\Rcolwidth}{%
2071           \initnumbering@sectcountR%
2072           \csuse{after@pendR@the\l@dpscR}%
2073         }%
2074         \ifdefstring{\columns@position}{R}{\hfill}%
2075       }%
2076     }%
2077     \global\csundef{after@pendL@the\l@dpscl}%
2078     \global\csundef{after@pendR@the\l@dpscR}%
2079   }%
2080   %

```

## XIX Parallel pages

This is considerably more complicated than parallel columns.

### XIX.1 Specific counters

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

```

2081 \newcount\numpagelinesL
2082 \newcount\numpagelinesR
2083 \newcount\l@dminpagelines
2084
2085 %

```

### XIX.2 Main macro

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

```

2086 \newcommand*{\Pages}{%
2087   \ifl@dpairing%

```

```

2088 \led@err@Pages@InsideEnv%
2089 \fi%
2090 \eledsection@correcting@skip=-2\baselineskip% line correcting for section
titles.
2091 \parledgroup@notespacing@set@correction%
2092 \typeout{}%
2093 \typeout{***** PAGES *****}%
2094 \ifnum\l@dnumstartsL=\l@dnumstartsR\else%
2095 \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}\the\l@dnumstartsR}%
2096 \fi%
2097 %

```

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

```

2098 \cleartol@devenpage%
2099 \l@dprintingpagestrue%
2100 \begingroup%
2101 %

```

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

```

2102 \setlength{\Lcolwidth}{\textwidth}%
2103 \setlength{\Rcolwidth}{\textwidth}%
2104 %

```

```

2105 \l@dzeropenalties%
2106 \endgraf\global\num@lines=\prevgraf%
2107 \global\num@linesR=\prevgraf%
2108 \global\par@line=\z@%
2109 \global\par@lineR=\z@%
2110 \global\l@dpscL=\z@%
2111 \global\l@dpscR=\z@%
2112 \writtenlinesLfalse%
2113 \writtenlinesRfalse%
2114 %

```

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

```

2115 \let\print@Xnotes\print@Xnotes@forpages%
2116 \let\print@notesX\print@notesX@forpages%
2117 %

```

Check if there are chunks to be processed.

```

2118 \check@pstarts%
2119 \loop\if@pstarts%
2120 %

```

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

```

2121 \global\advance\l@dpscL \@ne%
2122 \global\advance\l@dpscR \@ne%
2123 %

```

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

```

2124 \getlinesfromparlistL%
2125 \getlinesfromparlistR%
2126 \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2127 {\usernamecount{\l@dmaxlinesinpar\the\l@dpscL}}%
2128 \check@pstarts%
2129 \repeat%
2130 %

```

Zero the counts again, ready for the next bit.

```

2131 \global\l@dpscL=\z@%
2132 \global\l@dpscR=\z@%
2133 %

```

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

```

2134 \getlinesfrompagelistL%
2135 \getlinesfrompagelistR%
2136 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2137 {\l@dminpagelines}%
2138 %

```

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.

```

2139 \check@pstarts%
2140 \if@pstarts%
2141 %

```

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

```

2142 \global\advance\l@dpscL \@ne%
2143 \global\advance\l@dpscR \@ne%
2144 \restore@pstartL@pc%
2145 \restore@pstartR@pc%
2146 %

```

We have not processed any lines from these chunks yet, so zero the respective line counts.

```

2147 \global\@donereallinesL=\z@%
2148 \global\@donetotallinesL=\z@%
2149 \global\@donereallinesR=\z@%
2150 \global\@donetotallinesR=\z@%
2151 %

```

Start a loop over the boxes (chunks).

```

2152 \checkraw@text%
2153 %
2154 % \begingroup
2155 { \loop\ifaraw@text%
2156 %

```

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

```

2157 \checkpageL%
2158 \l@duselanguage{\theledlanguageL}%
2159 { \loop\ifl@dsamepage%
2160 %

```

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

```

2161 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2162 \csuse{before@pstartL@the\l@dpscl}%
2163 \global\csundef{before@pstartL@the\l@dpscl}%
2164 \do@lineL%
2165 \xifinlist{the\l@dpscl}{\eled@sections@@}
2166 {\print@eledsectionL}%
2167 {}%
2168 \advance\numpagelinesL \@ne%
2169 %

```

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

```

2170 \ifshiftedpstarts%
2171 \ifdim\ht\l@dleftbox>0pt%
2172 \parledgroup@correction@notespacing{L}%
2173 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2174 \else%
2175 \dimen0=\pagetotal%
2176 \advance\dimen0 by \baselineskip%
2177 \global\pagetotal=\dimen0%
2178 \fi%
2179 \else%
2180 \parledgroup@correction@notespacing{L}%
2181 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2182 \fi%
2183 %

```

Perhaps we have to move to the next (left) box. Check if we have got all we can onto the page. If not, repeat for the next line. Check if we have to print the optional argument

of the last pend. Check if the page is full. Check if the verse is split in two subsequent pages. Check there is any forced page breaks. Reset the verse skipnumber boolean

```

2184         \get@nextboxL%
2185     \global\l@dskipversenumberfalse%
2186         \ifprint@last@after@pendL%
2187             \csuse{after@pendL@the\l@dpscL}%
2188             \global\csundef{after@pendL@the\l@dpscL}%
2189         \fi%
2190         \checkpageL%
2191         \checkverseL%
2192         \checkpbL%
2193     \repeat%
2194 %

```

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.

```

2195         \ifl@dpagfull%
2196             \@writelinesonpageL{\the\numpagelinesL}%
2197         \else%
2198             \@writelinesonpageL{1000}%
2199         \fi%
2200 %

```

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.

```

2201     \numpagelinesL \z@%
2202     \parledgroup@correction@notespacing@init%
2203     \clearl@dleftpage }%
2204 %

```

Now do the same for the right text.

```

2205     \checkpageR%
2206     \l@duselanguage{\theledlanguageR}%
2207 {
2208     \loop\ifl@dsamepage%
2209         \initnumbering@sectcountR%
2210         \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2211         \csuse{before@pstartR@the\l@dpscR}%
2212         \global\csundef{before@pstartR@the\l@dpscR}%
2213         \do@lineR%
2214         \xifinlist{\the\l@dpscR}{\eled@sectionsR@}%
2215             {\print@eledsectionR}%
2216             {}%
2217         \advance\numpagelinesR \@ne%
2218         \ifshiftedpstarts%
2219             \ifdim\ht\l@drightbox>Opt%

```

```

2219         \parledgroup@correction@notespacing{R}%
2220         \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightrightbox}%
2221         \else%
2222             \dimen0=\pagetotal%
2223             \advance\dimen0 by \baselineskip%
2224             \global\pagetotal=\dimen0%
2225         \fi%
2226     \else%
2227         \parledgroup@correction@notespacing{R}%
2228         \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightrightbox}%
2229     \fi%
2230     \get@nextboxR%
2231     \global\l@dskipversenumberRfalse%
2232     \ifprint@last@after@pendR%
2233         \csuse{after@pendR@the\l@dpscR}%
2234         \global\csundef{after@pendR@the\l@dpscR}%
2235     \fi%
2236     \checkpageR%
2237     \checkverseR%
2238     \checkpbR%
2239     \repeat%
2240     \ifl@dpagfull%
2241         \@writelinesonpageR{\the\numpagelinesR}%
2242     \else%
2243         \@writelinesonpageR{1000}%
2244     \fi%
2245     \numpagelinesR=\z@%
2246     \parledgroup@correction@notespacing@init%
2247 %

```

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

```

2248     \clearl@drightrightpage}%
2249 %

```

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.

```

2250     \checkdraw@text%
2251     \ifaraw@text%
2252         \getlinesfrompagelistL%
2253         \getlinesfrompagelistR%
2254         \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2255             {\l@dminpagelines}%
2256     \fi%
2257     \repeat}%
2258 %

```

We have now output the text from all the chunks.

```

2259     \fi%
2260 %

```



Make sure that there are no inserts hanging around.

```
2261 \flush@notes%
2262 \flush@notesR%
2263 \endgroup%
2264 %
```

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

```
2265 \global\l@dpscL=\z@%
2266 \global\l@dpscR=\z@%
2267 \global\l@dnumpstartsL=\z@%
2268 \global\l@dnumpstartsR=\z@%
2269 \global\instanzaLfalse%
2270 \global\instanzaRfalse%
2271 \l@dprintingpagesfalse%
2272 \finish@Pages@notes%Needed to prevent final notes overlap line number
2273 \ignorespaces}
2274
2275
2276 %
```

### XIX.3 Ensure all notes be printed at the end of parallel pages

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

```
2277 \newcommand{\finish@Pages@notes}{%
2278 \def\do##1{%
2279 %
```

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

```
2280 \ifnocritical{%
2281 \global\newnamebox{##1footins}
2282 \fi
2283 \ifnofamiliar{%
2284 \global\newnamebox{footins##1}
2285 \fi
2286 %
```

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

```
2287 \ifvoid\csuse{##1footins}%
2288 \ifvoid\csuse{footins##1}\else%
2289 \newpage\null%
2290 \listbreak%
2291 \fi%
2292 \else%
2293 \newpage\null%
```

```

2294     \listbreak%
2295     \fi%
2296   }%
2297   \dolistloop{\@series}%
2298 }%
2299 %

```

#### XIX.4 Struts

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

```

\ledstrutR
2300 \newcommand*\ledstrutL{\strut}
2301 \newcommand*\ledstrutR{\strut}
2302
2303 %

```

#### XIX.5 Page clearing

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

```

2304 \providecommand{\cleartoevenpage}[1][\@empty]{%
2305   \clearpage
2306   \ifodd\c@page\hbox{ }#1\clearpage\fi}
2307
2308 \newcommand*\cleartol@evenpage{%
2309   \ifdim\pagetotal<\topskip% on an empty page
2310   \else
2311     \clearpage
2312   \fi
2313   \ifodd\c@page%
2314     \ifprevpgnotnumbered%
2315       \addtocounter{par@page}{-1}%
2316       \ifdef{\prevpgstyle}{\thispagestyle{\prevpgstyle}}{}%
2317     \fi%
2318     \hbox{ }\clearpage%
2319   \fi%
2320 }%
2321 %

```

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

```

2322 \newcommand*\clearl@dleftpage}{%
2323   \ifdim\pagetotal=0pt\hbox{}\fi%
2324   \newpage%
2325   \ifodd\c@page\else
2326     \led@err@LeftOnRightPage
2327     \hbox{}%
2328     \cleardoublepage
2329   \fi}
2330
2331 \newcommand*\clearl@drightpage}{%
2332   \ifdim\pagetotal=0pt\hbox{}\fi%
2333   \newpage%
2334   \ifodd\c@page
2335     \led@err@RightOnLeftPage
2336     \hbox{}%
2337     \cleartoevenpage
2338   \fi}
2339
2340 %

```

## XIX.6 Lines managing

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

```

2341 \newcommand*\getlinesfromparlistL}{%
2342   \ifx\linesinpar@listL\empty
2343     \gdef\@cs@linesinparL{0}%
2344   \else
2345     \gl@p\linesinpar@listL\to\@cs@linesinparL
2346   \fi}
2347 \newcommand*\getlinesfromparlistR}{%
2348   \ifx\linesinpar@listR\empty
2349     \gdef\@cs@linesinparR{0}%
2350   \else
2351     \gl@p\linesinpar@listR\to\@cs@linesinparR
2352   \fi}
2353
2354 %

```

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

```

2355 \newcommand*\getlinesfrompagelistL}{%
2356   \ifx\linesonpage@listL\empty
2357     \gdef\@cs@linesonpageL{1000}%
2358   \else

```

```

2359 \gl@p\linesonpage@listL\to\@cs@linesonpageL
2360 \fi}
2361 \newcommand*\getlinesfrompagelistR{%
2362 \ifx\linesonpage@listR\empty
2363 \gdef\@cs@linesonpageR{1000}%
2364 \else
2365 \gl@p\linesonpage@listR\to\@cs@linesonpageR
2366 \fi}
2367
2368 %

```

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

```

2369 \newcommand*\@writelinesonpageL}[1]{%
2370 \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2371 \next}
2372 \newcommand*\@writelinesonpageR}[1]{%
2373 \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2374 \next}
2375
2376 %

```

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

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

```

2377 \newcommand*\l@dcalc@maxoftwo}[3]{%
2378 \ifnum #2>#1\relax
2379 #3=#2\relax
2380 \else
2381 #3=#1\relax
2382 \fi}
2383 \newcommand*\l@dcalc@minoftwo}[3]{%
2384 \ifnum #2<#1\relax
2385 #3=#2\relax
2386 \else
2387 #3=#1\relax
2388 \fi}
2389
2390 %

```

## XIX.7 Page break managing

`\ifl@dsamepage` `\checkpageL` tests if the space and lines already taken on the page by text and footnotes is less than the constraints. If so, then `\ifl@dpagfull` is set FALSE and `\l@dsamepagetrue` `\ifl@dsamepage` is set TRUE. If the page is spatially full then `\ifl@dpagfull` is set TRUE and `\ifl@dsamepage` is set FALSE. If it is not spatially full but the maximum

```

\l@dsamepagefalse
\ifl@dpagfull
\l@dpagfulltrue
\l@dpagfullfalse
\checkpageL
\checkpageR

```

number of lines have been output then both `\ifl@dpagfull` and `\ifl@dsamepage` are set FALSE.

```

2391 \newif\ifl@dsamepage
2392   \l@dsamepagetrue
2393 \newif\ifl@dpagfull
2394
2395 \newcommand*\checkpageL}{%
2396   \l@dpagfulltrue
2397   \l@dsamepagetrue
2398   \check@goal
2399   \ifdim\pagetotal<\ledthegoal
2400     \ifnum\numpagelinesL<\l@dmminpagelines
2401       \else
2402         \l@dsamepagefalse
2403         \l@dpagfullfalse
2404       \fi
2405     \else
2406       \l@dsamepagefalse
2407       \l@dpagfulltrue
2408     \fi%
2409   \ifprint@last@after@pendL%
2410     \l@dpagfullfalse%
2411     \l@dsamepagefalse%
2412     \print@last@after@pendLfalse%
2413   \fi%
2414 }%
2415
2416 \newcommand*\checkpageR}{%
2417   \l@dpagfulltrue
2418   \l@dsamepagetrue
2419   \check@goal
2420   \ifdim\pagetotal<\ledthegoal
2421     \ifnum\numpagelinesR<\l@dmminpagelines
2422       \else
2423         \l@dsamepagefalse
2424         \l@dpagfullfalse
2425       \fi
2426     \else
2427       \l@dsamepagefalse
2428       \l@dpagfulltrue
2429     \fi%
2430   \ifprint@last@after@pendR%
2431     \l@dpagfullfalse%
2432     \l@dsamepagefalse%
2433     \print@last@after@pendRfalse%
2434   \fi%
2435 }%
2436
2437 %

```

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

```

2438 \newcommand{\checkpbL}{
2439   \IfStrEq{\led@pb@setting}{after}{
2440     \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@pagefulltrue\
l@dsamepagefalse}{
2441     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@pagefullfalse\
l@dsamepagetrue}{
2442   }}
2443   \IfStrEq{\led@pb@setting}{before}{
2444     \numdef{\next@absline}{\the\absline@num+1}
2445     \xifinlistcs{\next@absline}{l@prev@pb}{\l@pagefulltrue\
l@dsamepagefalse}{
2446     \xifinlistcs{\next@absline}{l@prev@nopb}{\l@pagefullfalse\
l@dsamepagetrue}{
2447   }}
2448 }
2449
2450 \newcommand{\checkpbR}{
2451   \IfStrEq{\led@pb@setting}{after}{
2452     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@pagefulltrue\
l@dsamepagefalse}{
2453     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@pagefullfalse\
l@dsamepagetrue}{
2454   }}
2455   \IfStrEq{\led@pb@setting}{before}{
2456     \numdef{\next@abslineR}{\the\absline@numR+1}
2457     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@pagefulltrue\
l@dsamepagefalse}{
2458     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@pagefullfalse\
l@dsamepagetrue}{
2459   }}
2460 }
2461 %

```

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

```

2462 \newcommand{\checkverseL}{
2463   \ifinstanzaL
2464   \iflednopbinverse
2465   \ifinserthangingsymbol
2466     \numgdef{\prev@abslineverse}{\the\absline@num-1}
2467     \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{
2468     \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\
prev@abslineverse}\fi}{
2469   \fi
2470   \fi
2471   \fi

```

```

2472 }
2473 \newcommand{\checkverse}{
2474 \ifinstanzaR
2475 \iflednopbinverse
2476 \ifinserthangingsymbolR
2477 \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2478 \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{
2479 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\
prev@abslineverse}\fi}{
2480 \fi
2481 \fi
2482 \fi
2483 }
2484 %

```

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

```

\check@goal
2485 \newdimen\ledthegoal
2486 \ifshiftedpstarts
2487 \newcommand*{\@goalfraction}{0.95}
2488 \else
2489 \newcommand*{\@goalfraction}{0.9}
2490 \fi
2491
2492 \newcommand*{\check@goal}{%
2493 \ledthegoal=\@goalfraction\pagegoal}
2494 \newcommand{\setgoalfraction}[1]{%
2495 \xdef\@goalfraction{#1}%
2496 }
2497 %

```

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

```

\ifwrittenlinesL
2498 \newif\ifwrittenlinesL
2499 \newif\ifwrittenlinesR
2500
2501 %

```

## XIX.8 Getting boxes content

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

```

2502 \newcommand*{\get@nextboxL}{%
2503 \ifvbox\namebox{1@dLcolrawbox\the1@dpsL}% box is not empty
2504 %

```

The current box is not empty; do nothing.

```

2505 \else%                                box is empty
2506 %

```

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

```

2507 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesL
2508 \parledgroup@notes@endL
2509 \else
2510 %

```

Sufficient lines have been output.

```

2511 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscl}=\@donetotallinesL
2512 \parledgroup@notes@endL
2513 \fi
2514 \ifwrittenlinesL\else
2515 %

```

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

```

2516 \@writelinesinparL
2517 \writtenlinesLtrue
2518 \fi
2519 \ifnum\l@dnumstartsL>\l@dpscl
2520 %

```

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

```

2521 \writtenlinesLfalse
2522 \ifbypstart@
2523 \global\line@num=0%
2524 \resetprevline@%
2525 \fi
2526 % Add the content of the optional argument of the previous \protect\cs{pend
2527 % \begin{macrocode}
2528 \csuse{after@pendL@\the\l@dpscl}%
2529 \global\csundef{after@pendL@\the\l@dpscl}%
2530 %

```

Check the number of lines

```

2531 \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscl}}%
2532 \the\@donetotallinesL}%
2533 \the\usernamecount{1@dmaxlinesinpar\the\l@dpscl}}%
2534 \global\@donetotallinesL \z@
2535 %

```

Go to the next pstart

```

2536 \global\advance\l@dpscl \@ne
2537 \global\pstartnumtrue%
2538 \restore@pstartL@pc%
2539 %

```



Add notes of parallel ledgroup.

```

2540     \parledgroup@notes@endL
2541     \parledgroup@correction@notespacing@final{L}
2542     \else
2543 %

2544     \print@last@after@pendLtrue%
2545     \fi
2546     \fi
2547 \fi}
2548 %

2549 \newcommand*{\get@nextboxR}{%
2550 \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2551 \else% box is empty
2552 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2553 \parledgroup@notes@endR
2554 \else
2555 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2556 \parledgroup@notes@endR
2557 \fi
2558 \ifwrittenlinesR\else
2559 \@writelinesinparR
2560 \writtenlinesRtrue
2561 \fi
2562 \ifnum\l@dnumstartsR>\l@dpscR
2563 \writtenlinesRfalse
2564 \ifbypstartR
2565 \global\line@numR=0%
2566 \resetprevline%
2567 \fi
2568 \csuse{after@pendR@the\l@dpscR}%
2569 \global\csundef{after@pendR@the\l@dpscR}%
2570 \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2571 {\the\@donetotallinesR}%
2572 {\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2573 \global\@donetotallinesR \z@
2574 \global\advance\l@dpscR \@ne
2575 \global\pstartnumRtrue%
2576 \restore@pstartR@pc%
2577 \parledgroup@notes@endR
2578 \parledgroup@correction@notespacing@final{R}
2579 \else
2580 \print@last@after@pendRtrue%
2581 \fi
2582 \fi
2583 \fi}
2584
2585 %

```

## XX Page numbering

The `sameparallelpagnumber` option allows the same page number on both left and right side. The `prevpnotnumbered` option allows an empty (not numbered) right-side page before `\Pages`.

We cannot implement these two options by changing the value of the page counter, since its value is used by many  $\TeX$  features to determine whether a page is left (even-numbered) or right (odd-numbered). Consequently, we have to do it by patching `\thepage`, in order to use the value of the `par@page` counter instead of value of page counter.

This counter will be increased in a patched version of the  $\TeX$ 's `\@outputpage` macro, as is the page counter in this macro. However, this increase will take account of the options.

`\par@patch@thepage` `\par@patch@thepage` patches `\thepage` in order to use the value of `par@page` counter and not the value of `par@page`. It must be called after any redefinition of `\thepage`. That why we insert it at the end of the  $\TeX$  macro `\pagenumbering`, which is called by some `\xxxmatter` commands. In the case of `memoir` class using, we insert it at the end of `\@mempnum`. When using `\pagenumbering`, we also need to restart `par@page` counter. Consequently, we have wrapped `\par@patch@thepage` and counter restart in `\par@patch@pagenumbering`. We also call `\par@patch@thepage` it at the beginning of the document.

```

2586
2587 \newcommand{\par@patch@thepage}{%
2588   \ifboolexpr{%
2589     bool{sameparallelpagnumber}%
2590     or bool{prevpnotnumbered}%
2591   }%
2592   {%
2593     \patchcmd{\thepage}%
2594       {page}{par@page}%
2595       {}%
2596       {\led@error@fail@patch@thepage}%
2597   }{}%
2598 }%
2599
2600 \newcommand{\par@patch@pagenumbering}{%
2601   \ifboolexpr{%
2602     bool{sameparallelpagnumber}%
2603     or bool{prevpnotnumbered}%
2604   }%
2605   {%
2606     \setcounter{par@page}{1}%
2607   }%
2608   {}%
2609   \par@patch@thepage%
2610 }%
2611

```

```

2612 \ifl@dmemoir%
2613   \apptocmd{\@mempnum}%
2614     {\par@patch@pagenumbering}%
2615     {}%
2616     {\led@error@fail@patch@mempnum}%
2617
2618 \else%
2619   \apptocmd{\pagenumbering}%
2620     {\par@patch@pagenumbering}%
2621     {}%
2622     {\led@error@fail@patch@pagenumbering}%
2623 \fi%
2624
2625 \AtBeginDocument{\par@patch@thepage}%
2626 %

```

`\@outputpage` As its name says, `\@outputpage` is a  $\LaTeX$ 's macro called in the output routine. It is this macro which increases the page counter.. We patch it in order to increase, conditionally, the `par@page` counter.

```

2627 \AtBeginDocument{%
2628   \apptocmd{\@outputpage}{%
2629     \ifsameparallelpagenumber%
2630       \ifl@dprintingpages%
2631         \ifodd\c@page\else%
2632           \stepcounter{par@page}%
2633           \fi%
2634         \else%
2635           \stepcounter{par@page}%
2636           \fi%
2637         \else%
2638           \stepcounter{par@page}%
2639           \fi%
2640       }%
2641     }%
2642     {\led@error@fail@patch@outputpage}%
2643 }
2644 %

```

`\thepar@page` And now, initialize `par@page` counter.

```

2645 \newcounter{par@page}%
2646 \setcounter{par@page}{1}%
2647 %

```

## XXI Sections' titles' commands

As switching from left to right pages does not clear the page since v1.13.0, but only creates new pages, no `\vbox{}` is inserted, and consequently parallel chapters are mis-

aligned.

So we patch the `\chapter` command in order to prevent this problem.

```
\chapter 2648 \pretocmd{\chapter}{%
2649   \ifl@dprintingpages%
2650   \vbox{}}%
2651   \fi%
2652 }%
2653 {}%
2654 {}%
2655 %
```

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

```
2656 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2657 \eledsectnotoc{R}
2658 %
```

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

```
2659 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2660 \eledsectmark{L}
2661 %
```

`\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`.

```
2662 \newskip\eledsection@correcting@skip
2663 %
```

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

```
2664 \newwrite\eled@sectioningR@out
2665 %
```

## XXII Page break/no page break, depending on the specific line

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

`\prev@pbR` The `\l@prev@pbR` macro is a `etoolbox`'s 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).

```

2666 \def\l@prev@pbR{}
2667 \def\l@prev@nopbR{}
2668 %

```

`\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.

```

2669 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2670 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2671 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2672 \newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\led@nopbnumR{#1}}}
2673 %

```

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

```

2674 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2675 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}
2676 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2677 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}
2678 %

```

## XXIII Parallel ledgroup

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

```

2679 \newmarks\parledgroup@
2680 \newmarks\parledgroup@series
2681 \newmarks\parledgroup@type
2682 %

```

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

```

2683 \newcommand{\parledgroup@notes@startL}{%
2684 \ifnum\usenamecount{l@dmxlinesinpar\the\l@dpscl}>0%
2685 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{}%
2686 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{}%
2687 \fi%

```

```

2688 \global\ledgroupnotesL@true%
2689 \insert@noterule@ledgroup{L}%
2690 }
2691 \newcommand{\parledgroup@notes@startR}{%
2692 \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}>0%
2693 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{%
2694 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{%
2695 \fi%
2696 \global\ledgroupnotesR@true%
2697 \insert@noterule@ledgroup{R}%
2698 }
2699 %

```

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

```

2700 \newcommand{\parledgroup@notes@endL}{%
2701 \global\ledgroupnotesL@false%
2702 }
2703 \newcommand{\parledgroup@notes@endR}{%
2704 \global\ledgroupnotesR@false%
2705 }
2706 %

```

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

```

2707 \newcommand{\insert@noterule@ledgroup}[1]{
2708 \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2709 \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2710 \csuse{ifledgroupnotes#1@}
2711 \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2712 \csuse{\splitbotmarks\parledgroup@series footnoterule}
2713 \fi
2714 }
2715 {}
2716 \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2717 \csuse{ifledgroupnotes#1@}
2718 \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2719 \csuse{footnoterule\splitbotmarks\parledgroup@series}
2720 \fi
2721 }{}
2722 }
2723 {}
2724 }
2725 %

```

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

```
2726 \newcommand{\setparledgroupnotespacing}[1]{\gdef\@parledgroupnotespacing
2727 {#1}}
2728 \newcommand{\@parledgroupnotespacing}{}
2729 %
```

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

```
2729 \dimdef{\parledgroup@notespacing@correction}{0pt}
2730 \newcommand{\parledgroup@notespacing@set@correction}{%
2731 {\@getfirstseries\csuse{Xnotefontsize@\@firstseries}%We suppose all the
2732 series has the same footnote size setup
2733 \@parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
2734 \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing
2735 }%
2736 }
2737 %
```

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

```
2736 \newcommand{\parledgroup@correction@notespacing@init}{
2737 \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2738 \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2739 }
2740 \parledgroup@correction@notespacing@init
2741 %
```

`\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 is called after the print of each `pstart/pend`.

```
2742 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2743 \ifparledgroup
2744 \vspace{\parledgroup@notespacing@correction@accumulated}
2745 \parledgroup@correction@notespacing@init%
2746 \ifstrequal{#1}{L}{
2747 \numdef{\@checking}{\the\l@dpscL-1}
2748 }{
2749 \numdef{\@checking}{\the\l@dpscR-1}
2750 }
2751 \dimdef{\@beforenotes@current@diff}{\csuse{\parledgroup@beforenotes@
2752 @checking L}-\csuse{\parledgroup@beforenotes@\@checking R}}%
2753 \ifstrequal{#1}{L}%
```

```

2753     {% Left
2754     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\
@beforenotes@current@diff}}%
2755     }%
2756     {% Right
2757     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\
@beforenotes@current@diff}}{}
2758     }%
2759     \fi
2760 }
2761 %

```

`\parledgroup@correction@notespacing` `\parledgroup@correction@notespacing` is used before each printed line. If it is 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` and `\parledgroup@notespacing@correction@modulo`. If `\parledgroup@notespacing@correction` 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.

```

2762 \newcommand{\parledgroup@correction@notespacing}[1]{%
2763     \csuse{ifledgroupnotes#1@}%
2764     \vspace{-\parledgroup@notespacing@correction}%
2765     \dimdef{\parledgroup@notespacing@correction@accumulated}{\
parledgroup@notespacing@correction@accumulated+\
parledgroup@notespacing@correction}%
2766     \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo+\
parledgroup@notespacing@correction}%
2767     \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip
}{\advance\numpagelinesL -\@ne%
2768     \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo-\baselineskip}%
2769     }% mean greater than equal
2770     \fi%
2771 }
2772 %

```

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

```

2773 \dimdef\parledgroup@beforenotesL{0pt}
2774 \dimdef\parledgroup@beforenotesR{0pt}
2775 %

```



`\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.

```

2776 \newcommand{\parledgroup@beforenotes@save}[1]{
2777   \ifparledgroup
2778     \csdimgdef{@parledgroup@beforenotes@the\csuse{1@dnumstarts#1}#1}{\
csuse{parledgroup@beforenotes#1}}
2779     \csdimgdef{parledgroup@beforenotes#1}{0pt}
2780   \fi
2781 }
2782 %

```

## XXIV Compatibility with eledmac

Here, we define some command for the eledmac-compat option.

```

2783 \ifeledmaccompat%
2784
2785
2786   \unless\ifnocritical@
2787   \let\onlyXside\Xonlyside
2788   \fi
2789 \fi
2790 %

```

## XXV The End

</code>

## Appendix A Some things to do when changing version

### Appendix A.1 Migration to eledpar 1.4.3

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

However, if you want to return to automatic flushright margins for verses with hanging indents, you have to redefine the `\hangingsymbol` command.

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

See the following two examples:

With standard `\hangingsymbol`:

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

With the modification of the `hangingsymbol`:

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

### Appendix A.2 Migration from eledpar to reledpar

As for migration from eledmac to reledmac:

- One option has been removed because it is deprecated.
- Some of the customizations previously made by `\renewcommand` have been replaced with commands.
- Some command names have been changed in order to have a more logical and uniform pattern.

#### Appendix A.2.1 Deprecated options

The `shiftedverses` option has been removed. Use the general `shiftedpstart` option instead.

#### Appendix A.2.2 `\renewcommand` replaced with command

Many uses of `\renewcommand` have been replaced with uses of specific commands. Please read the handbook about these particular commands.

<i>Deprecated <code>\renewcommand</code></i>	<i>Replaced with</i>
<code>\goalfraction</code>	<code>\setgoalfraction</code>
<code>\parledgroupnotespacing</code>	<code>\setparledgroupnotespacing</code>
<code>\Rlineflag</code>	<code>\setRlineflag</code>

### Appendix A.2.3 Commands the names of which have changed

In order to ease the migration from eledpar to reledpar, you may load reledmac with eledmac-compat option. However, it is advised to change the command names.

<i>Old command</i>	<i>New command</i>
<code>\onlyXside</code>	<code>\Xonlyside</code>

### Appendix A.3 Migration to reledpar 2.2.0

The *astanza* can take now an option argument. Consequently, if the first line of verse in a *astanza* environment starts with brackets [], you must precede them with a `\relax`. If you do not do it, the content of the brackets will be considered as an optional argument of the *astanza* environment.

### Appendix A.4 Migration to reledmac 2.3.0

The line number style (alphabetic, numeric, etc.) for the notes of the right side are now defined by the value you set to `\linenumberstyleR` or `\linenumberstyle*`, and not by the value you set to `\linenumberstyle` which is kept for left side.

The same is true for sub-line number styles and `\sublinenumberstyleR` or `\sublinenumberstyle*`, which are distinct from `\sublinenumberstyle`.

Consequently, if you have changed line number representation in footnotes with `\linenumberstyle` and `\sublinenumberstyle`, check your settings for these control sequences.

## 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

### Symbols

<code>\@adv</code> .....	1
<code>\@astanza@line</code> .....	1
<code>\@cs@linesinparL</code> .....	1

<code>\@cs@linesinparR</code> .....	1
<code>\@cs@linesonpageL</code> .....	1
<code>\@cs@linesonpageR</code> .....	1
<code>\@donereallinesL</code> .....	1
<code>\@donereallinesR</code> .....	1
<code>\@donetotallinesL</code> .....	1
<code>\@donetotallinesR</code> .....	1
<code>\@eledsectionL</code> .....	1
<code>\@eledsectionR</code> .....	1
<code>\@lab</code> .....	1
<code>\@lopL</code> .....	1
<code>\@lopR</code> .....	1
<code>\@nl</code> .....	1
<code>\@nl@regR</code> .....	1
<code>\@outputpage</code> .....	1
<code>\@parledgroupnotespacing</code> .....	1
<code>\@pend</code> .....	1
<code>\@pendR</code> .....	1
<code>\@pstartsfalse</code> .....	1
<code>\@pstartstrue</code> .....	1
<code>\@ref</code> .....	1
<code>\@ref@regR</code> .....	1
<code>\@set</code> .....	1
<code>\@stopastanza</code> .....	1
<code>\@writelinesinparL</code> .....	1
<code>\@writelinesinparR</code> .....	1
<code>\@writelinesonpageL</code> .....	1
<code>\@writelinesonpageR</code> .....	1
CLASSmemoir .....	98
COMMAND\@Rlineflag .....	61, 66
COMMAND\@adv .....	31, 130
COMMAND\@cs@linesinparL .....	91
COMMAND\@cs@linesonpageL .....	91
COMMAND\@eledsectionL .....	75
COMMAND\@eledsectionR .....	75
COMMAND\@eledsectmark .....	100
COMMAND\@eledsectnotoc .....	100
COMMAND\@footnotetext .....	67
COMMAND\@goalfraction .....	9, 95
COMMAND\@l@dtempcnta .....	59
COMMAND\@lab .....	30, 66, 130
COMMAND\@lopL .....	35, 86, 92
COMMAND\@lopR .....	35, 92
COMMAND\@mempnum .....	98
COMMAND\@namedef .....	70
COMMAND\@namuse .....	70
COMMAND\@nl .....	30, 36, 66, 130
COMMAND\@nl@regR .....	30
COMMAND\@outputpage .....	98, 99
COMMAND\@page .....	66

COMMAND\@parledgroupnotespacing	103
COMMAND\@pend	35
COMMAND\@pendR	35
COMMAND\@pstartstrue	74
COMMAND\@ref	33, 34, 36, 130
COMMAND\@ref@regR	33
COMMAND\@set	31, 130
COMMAND\@sw	34
COMMAND\AtBeginPairs	7, 41, 128
COMMAND\AtEveryPend	128–130
COMMAND\AtEveryPstart	2, 13, 47, 76, 128–130
COMMAND\AtEveryPstartCall	2, 13, 47, 129
COMMAND\Clear the right lines for \read@linelist	29
COMMAND\Columns	6, 7, 13, 40, 75, 79, 125–128, 130, 131
COMMAND\Columns@print@after@pend	82
COMMAND\Columns@print@before@pstart	82
COMMAND\Lcolwidth	7, 9, 84
COMMAND\Leftsidehook	125
COMMAND\Leftsidehookend	125
COMMAND\Pages	3, 6, 8, 9, 13, 40, 57, 61, 62, 83, 84, 89, 98, 103, 125, 128–131
COMMAND\Rcolwidth	7, 9, 84
COMMAND\Rightsidehook	125
COMMAND\Rightsidehookend	125
COMMAND\Rlineflag	106
COMMAND\Xmaxhnotes	10
COMMAND\Xnoteswidthliketwocolumns	8, 128
COMMAND\Xonlyside	10, 61, 107
COMMAND&	14
COMMAND\absline@numR	29
COMMAND\add@penalties	59
COMMAND\add@penaltiesL	59
COMMAND\advanceline	31, 36, 130
COMMAND\affixline@num	55
COMMAND\affixline@numR	55, 125, 126
COMMAND\affixpstart@num	57
COMMAND\affixpstart@numR	57
COMMAND\affixside@note	66
COMMAND\aftercolumnseparator	8, 79, 127
COMMAND\araw@textfalse	74
COMMAND\araw@texttrue	74
COMMAND\at@begin@pairs	41
COMMAND\autopar	13
COMMAND\ballast@count	59
COMMAND\baselineskip	79, 104
COMMAND\bbl@set@language	71, 72, 130
COMMAND\beforecolumnseparator	8, 79, 127
COMMAND\begin	14
COMMAND\beginnumbering	11–13, 22, 30, 80, 126, 127, 130
COMMAND\beginnumberingR	36
COMMAND\bf	126

COMMAND\bfseries	126
COMMAND\brokenpenalty	59
COMMAND\chapter	100, 125
COMMAND\check@goal	95
COMMAND\check@pstarts	74
COMMAND\checkpageL	86, 92
COMMAND\checkpb@columns	78
COMMAND\checkpbL	94
COMMAND\checkpbR	94
COMMAND\checkraw@text	74
COMMAND\checkverseL	94
COMMAND\checkverseR	94
COMMAND\clear(double)page	90
COMMAND\clearl@dleftpage	90
COMMAND\clearl@drightpage	90
COMMAND\clearpage	90, 129
COMMAND\cleartoevenpage	90
COMMAND\cleartol@devenpage	90
COMMAND\columnrulewidth	7, 79
COMMAND\columns@position	79
COMMAND\columnseparator	7
COMMAND\columnsposition	7, 127
COMMAND\correct@Xfootins@box	62, 64, 129
COMMAND\correct@footinsX@box	62, 129
COMMAND\critext	129
COMMAND\curname	40
COMMAND\displaywidowpenalty	59
COMMAND\do@actions	53
COMMAND\do@actions@fixedcode	125
COMMAND\do@actions@nextR	53
COMMAND\do@actionsR	53, 125
COMMAND\do@ballast	59
COMMAND\do@ballastR	53
COMMAND\do@insidelineLhook	127
COMMAND\do@insidelineRhook	127
COMMAND\do@line	48
COMMAND\do@line(L/R)	51
COMMAND\do@lineL	48, 59, 125, 126
COMMAND\do@lineLhook	125
COMMAND\do@lineR	51, 125–127
COMMAND\do@lineRhook	125
COMMAND\do@lockoff	131
COMMAND\do@lockoffR	32
COMMAND\do@lockon	130
COMMAND\do@lockonR	32
COMMAND\doinsidelineLhook	128
COMMAND\doinsidelineRhook	128
COMMAND\dolineLhook	128
COMMAND\dolineRhook	128
COMMAND\edindex	129

COMMAND\edlabel	126, 129
COMMAND\edtext	33, 36, 37, 128, 129
COMMAND\eled@sectioningR@out	100
COMMAND\eledchapter	129
COMMAND\eledsection	128, 129, 131
COMMAND\eledsection@correcting@skip	100
COMMAND\eledsectmark	16, 100
COMMAND\eledsectnotoc	16, 100
COMMAND\eledxxx	128
COMMAND\end	14
COMMAND\endgraf	46
COMMAND\endlock	37, 130
COMMAND\endnumbering	11, 13, 22, 130
COMMAND\endsub	36, 130
COMMAND\endumbering	11
COMMAND\expandafter	38
COMMAND\extensionchars	21
COMMAND\firstlinenum	12, 127, 131
COMMAND\firstsublinenum	12, 127, 131
COMMAND\fix@page	31, 130
COMMAND\flag@end	36, 125, 128
COMMAND\flag@start	36, 128
COMMAND\flush@notesR	60
COMMAND\footnoteX	38
COMMAND\footnoteXmk	10
COMMAND\footnoteXnomk	10, 38
COMMAND\frontmatter	17
COMMAND\get@nextboxL	126
COMMAND\get@nextboxR	126
COMMAND\getline@numL	52
COMMAND\getline@numR	52
COMMAND\getlinesfrompagelistL	91
COMMAND\getlinesfrompagelistR	91
COMMAND\getlinesfromparlistL	91
COMMAND\getlinesfromparlistR	91
COMMAND\gl@p	38
COMMAND\goalfraction	106
COMMAND\hangingsymbol	106, 126
COMMAND\hfill	78, 79
COMMAND\hidenumbering	12, 130
COMMAND\ifbypage@	130
COMMAND\ifbypstart@R	130
COMMAND\ifdim	78
COMMAND\ifinserthangingsymbol	68
COMMAND\ifinserthangingsymbolR	68
COMMAND\ifl@dpagefull	92, 93
COMMAND\ifl@dpaging	19, 128
COMMAND\ifl@dpairing	19, 125
COMMAND\ifl@dsamelang	127
COMMAND\ifl@dsamepage	92, 93

COMMAND\ifl@pagefull	86
COMMAND\ifledRcol	19
COMMAND\iflledRcol	126
COMMAND\ifnumberedpar@	43
COMMAND\ifnumberingR	126
COMMAND\ifnumberpstart	40
COMMAND\ifpst@rtedL	21, 22, 44, 125
COMMAND\ifpst@rtedR	21
COMMAND\ifsublines@	31
COMMAND\insert@countR	33
COMMAND\insert@noterule@ledgroup	102
COMMAND\insertlines@list	33
COMMAND\insertlines@listR	33
COMMAND\inserts@list	43
COMMAND\inserts@listR	58
COMMAND\l@d@nums	37, 61
COMMAND\l@d@set	31, 37, 130
COMMAND\l@dLcolrawbox	43
COMMAND\l@dLcolrawbox1	73
COMMAND\l@dLcolrawbox2	73
COMMAND\l@dRcolrawbox	43
COMMAND\l@dbfnote	67, 130
COMMAND\l@dcalc@maxoftwo	92
COMMAND\l@dcalc@minoftwo	92
COMMAND\l@dchecklang	125, 128
COMMAND\l@dcsnote	127
COMMAND\l@dleftbox	48, 86, 129
COMMAND\l@dlinenumR	28, 125
COMMAND\l@dlsnote	127
COMMAND\l@dmake@labels	66
COMMAND\l@dmaxlinesinpar	85
COMMAND\l@dmaxlinesinpar1	73
COMMAND\l@dminpagelines	85, 125
COMMAND\l@dnumpstartsL	73, 125
COMMAND\l@dprintingcolumnstrue	129
COMMAND\l@dprintingpagestrue	129
COMMAND\l@dpscL	76, 84, 85, 96
COMMAND\l@dpscR	76, 84, 85
COMMAND\l@drsnote	127
COMMAND\l@dsetupmaxlinecounts	73
COMMAND\l@duselanguage	71, 72, 125
COMMAND\l@dzeromaxlinecounts	73
COMMAND\l@prev@nopbR	100
COMMAND\l@prev@pbR	100
COMMAND\labelpstarttrue	126
COMMAND\labelref@list	66
COMMAND\labelref@listR	65
COMMAND\lang	71
COMMAND\last@page@numR	31
COMMAND\led	126



COMMAND\led@nopbR	101
COMMAND\led@nopbnumR	101
COMMAND\led@pbR	101
COMMAND\led@pbnumR	101
COMMAND\ledinnerrote	15
COMMAND\ledleftnote	15
COMMAND\lednopb	15, 78, 94
COMMAND\lednopbR	101
COMMAND\lednopbnumR	101
COMMAND\ledouterote	15
COMMAND\ledpb	78, 94
COMMAND\ledpbR	101
COMMAND\ledpbnumR	101
COMMAND\ledrightnote	15
COMMAND\ledsidenote	15
COMMAND\ledstrutL	125
COMMAND\ledstrutR	125
COMMAND\ledthegoal	95
COMMAND\ledtrutL	125
COMMAND\leftlinenumR	28, 125
COMMAND\let	38
COMMAND\line@list@R	34
COMMAND\line@list@stuff	30, 36
COMMAND\line@margin	26
COMMAND\line@marginR	26, 125
COMMAND\line@numR	29
COMMAND\lineation	12, 129
COMMAND\lineation*	12, 25, 128
COMMAND\lineationR	12, 25, 129
COMMAND\linenum@out	66
COMMAND\linenum@outR	35
COMMAND\linenumberstyle	12, 107
COMMAND\linenumberstyle*	107
COMMAND\linenumberstyleR	12, 107
COMMAND\linenumincrement	12, 127, 131
COMMAND\linenummargin	12, 25, 125, 130
COMMAND\linenumrepR	27, 125
COMMAND\linesinpar@listL	35, 91
COMMAND\linesonpage@listL	35, 91
COMMAND\lock@off	32
COMMAND\lock@on	32
COMMAND\mainmatter	17
COMMAND\makeatletter	50
COMMAND\maxchunks	6, 14, 73, 74
COMMAND\maxhnotesX	10
COMMAND\memorydump	11, 24
COMMAND\n@num	129
COMMAND\new@lineL	36
COMMAND\new@lineR	36
COMMAND\newhookcommand@series	39

COMMAND\newif	129
COMMAND\newpage	89, 90, 129
COMMAND\newseries	39
COMMAND\newseries@	37
COMMAND\newseries@par	37, 39
COMMAND\noledxxx	128
COMMAND\nomark@	38
COMMAND\normalbfnoteX	125, 130
COMMAND\notesXwidthliketwocolumns	8, 128
COMMAND\num@lines	59
COMMAND\num@lines(R)	43
COMMAND\numberingR	23
COMMAND\numberlinefalse	5
COMMAND\numberonlyfirstinline	126
COMMAND\numberpstartfalse	12
COMMAND\numberpstarttrue	12, 126, 131
COMMAND\one@line	43, 68
COMMAND\one@lineR	43
COMMAND\onlyXside	107
COMMAND\onlysideX	10, 61, 130
COMMAND\otherlanguage	131
COMMAND\page@action	31, 130
COMMAND\pagenumbering	98, 131
COMMAND\pagetotal	86, 129
COMMAND\par@line	59
COMMAND\par@line(R)	43
COMMAND\par@patch@pagenumbering	98
COMMAND\par@patch@thepage	98
COMMAND\parledgroup@	101
COMMAND\parledgroup@beforenotes@save	105
COMMAND\parledgroup@beforenotesL	104
COMMAND\parledgroup@beforenotesR	104
COMMAND\parledgroup@correction@notespacing	104
COMMAND\parledgroup@correction@notespacing@final	103
COMMAND\parledgroup@correction@notespacing@init	103
COMMAND\parledgroup@notes@endL	102
COMMAND\parledgroup@notes@endR	102
COMMAND\parledgroup@notes@startL	101
COMMAND\parledgroup@notes@startR	101
COMMAND\parledgroup@notespacing@correction	103, 104
COMMAND\parledgroup@notespacing@correction@accumulated	104
COMMAND\parledgroup@notespacing@correction@modulo	104
COMMAND\parledgroup@notespacing@set@correction	103
COMMAND\parledgroup@series	101
COMMAND\parledgroup@type	101
COMMAND\parledgroupnotespacing	106
COMMAND\parledgrouptrue	15
COMMAND\patchcmd	130
COMMAND\pausenumbering	23, 24
COMMAND\pend	3, 6, 13–15, 40, 43, 46–48, 72, 82, 127, 128, 130, 131

COMMAND\pendL	127, 128
COMMAND\pendR	128
COMMAND\pends	13
COMMAND\prev@nopbR	101
COMMAND\prev@pbR	101
COMMAND\prevpgstyle	19
COMMAND\print@Xnotes	61
COMMAND\print@Xnotes@forpages	61, 129
COMMAND\print@columnseparator	78, 128
COMMAND\print@eledsectionL	50
COMMAND\print@line	49
COMMAND\print@lineL	49
COMMAND\print@notesX@forpages	129
COMMAND\printlines	61
COMMAND\printlinesR	61, 125
COMMAND\pstart	3, 6, 12–15, 24, 37, 40, 43, 44, 47, 72, 76, 82, 126, 127, 130, 131
COMMAND\pstartL	47, 127
COMMAND\pstartR	47, 126, 127
COMMAND\pstartinfootnote	129
COMMAND\raw@text	72
COMMAND\read@linelist	29, 30, 109, 130
COMMAND\ref@reg	33
COMMAND\ref@regR	33, 130
COMMAND\relax	107
COMMAND\reledmac	130
COMMAND\renewcommand	106
COMMAND\resumenumbering	23, 24, 127
COMMAND\resumenumberingR	128
COMMAND\rightlinenumR	28, 125
COMMAND\section	125
COMMAND\section@num	21
COMMAND\selectlanguage	13, 71, 72
COMMAND\set@line	37, 130
COMMAND\set@line@action	31, 130
COMMAND\setRlineflag	13, 106
COMMAND\setgoalfraction	9, 106
COMMAND\sethangingsymbol	15
COMMAND\setline	31, 36, 130
COMMAND\setlinenum	31, 37, 130
COMMAND\setnoteposition...	80
COMMAND\setparledgroupnotespacing	106, 131
COMMAND\setposition...	80
COMMAND\setprintlines	125
COMMAND\setstanzaindents	7, 14, 106
COMMAND\setwidth...	80
COMMAND\sidenotemargin	15, 128
COMMAND\sidenotemargin*	15, 128
COMMAND\skipnumbering	12, 129
COMMAND\sloppy	7
COMMAND\stanza	6, 7, 12, 14, 41, 68, 126

COMMAND\stanzanumtrue	15
COMMAND\startlock	37, 130
COMMAND\startsub	36, 130
COMMAND\sub@action	31, 131
COMMAND\sub@off	66
COMMAND\sub@on	66
COMMAND\subline@numR	29
COMMAND\sublinenumberstyle	12, 107
COMMAND\sublinenumberstyle*	107
COMMAND\sublinenumberstyleR	12, 107
COMMAND\sublinenumincrement	12, 127, 131
COMMAND\sublinenumrepR	27, 125
COMMAND\sza@0@	14
COMMAND\textheight	10
COMMAND\textwidth	41
COMMAND\thefootnoteX	127
COMMAND\theledlanguageL	71, 72
COMMAND\theledlanguageR	71, 72
COMMAND\thepage	17, 98
COMMAND\thepstartL	12, 126
COMMAND\thepstartR	12, 126
COMMAND\thestanzaL	15
COMMAND\thestanzaR	15
COMMAND\vbox	45
COMMAND\vl@dbfnote	67
COMMAND\vskip	102
COMMAND\vsplit	59
COMMAND\widthliketwocolumns	8, 19
COMMAND\widthliketwocolumnsfalse	8
COMMAND\widthliketwocolumnstrue	8
COMMAND\xright@appenditem	38
COMMAND\xspace	18
COMMAND\xxxfootstart	80
COMMAND\xxxmatter	98
ENVIRONMENTLeftside	42
ENVIRONMENTRightside	42
ENVIRONMENTastanza	14, 68, 69, 107, 131
ENVIRONMENTpages	41
ENVIRONMENTpairs	41
PACKAGEEDMAC	107
PACKAGEEDSTANZA	107
PACKAGEEledmac	37, 61, 129
PACKAGEEledpar	129
PACKAGETABMAC	107
PACKAGEbabel	13, 14, 70–72
PACKAGEedmac	107
PACKAGEeledmac	4, 73, 105–107, 127, 128, 130
PACKAGEeledpar	4, 5, 10, 26, 106, 107, 127–129
PACKAGEetoolbox	78, 100
PACKAGEledmac	5

PACKAGEledpar	1, 5
PACKAGEMemoir	107
PACKAGEmusixtex	127
PACKAGEpolyglossia	13, 70–72
PACKAGEledmac	1, 3–6, 8–10, 12–16, 18, 19, 21, 22, 26, 28–33, 35–40, 49, 57, 66–68, 84, 89, 106, 107, 130, 131
PACKAGEreledpar	1, 3–6, 8, 9, 14, 15, 17–19, 24, 28, 29, 35, 37, 39, 40, 66, 106, 107, 130
PACKAGEsetspace	2, 16
PACKAGEXkeyval	18

## A

\absline@numR	1
\actionlines@listR	1
\actions@listR	1
\add@inserts@nextR	1
\add@insertsR	1
\add@penaltiesL	1
\add@penaltiesR	1
\advanceline	1
\affixline@numR	1
\affixpstart@numL	1
\affixpstart@numR	1
\affixside@noter	1
\aftercolumnseparator	1, 8
\araw@textfalse	1
\araw@texttrue	1
astanza (environment)	14
\AtBeginPairs	1, 7
\AtEveryPstartCall	1
\autopar	13

## B

\bbl@set@language	1
\beforecolumnseparator	1, 8
\beginnumbering	11
\beginnumberingR	1

## C

\c@firstlinenumR	1
\c@firstsublinenumR	1
\c@linenumincrementR	1
\c@sublinenumincrementR	1
\ch@ck@l@ckR	1
\ch@cksub@l@ckR	1
\chapter	1
\chapterinpages	1
\check@goal	1
\check@pstarts	1
\checkpageL	1
\checkpageR	1

<code>\checkpb@columns</code>	1
<code>\checkpbL</code>	1
<code>\checkpbR</code>	1
<code>\checkraw@text</code>	1
<code>\checkverseL</code>	1
<code>\checkverseR</code>	1
<code>\clearl@dleftpage</code>	1
<code>\clearl@drightpage</code>	1
<code>\cleartoevenpage</code>	1
<code>\cleartol@devenpage</code>	1
<code>\columnrulewidth</code>	1, 7
<code>\Columns</code>	1, 6
<code>\columns@position</code>	1
<code>\Columns@print@after@pend</code>	1
<code>\Columns@print@before@pstart</code>	1
<code>\columnseparator</code>	1, 7
<code>\columnspan</code>	1, 7
<code>\correct@footinsX@box</code>	1
<code>\correct@Xfootins@box</code>	1
<code>\countLline</code>	1
<code>\countRline</code>	1
<code>\critext</code>	1

## D

<code>\do@actions@fixedcodeR</code>	1
<code>\do@actions@nextR</code>	1
<code>\do@actionsR</code>	1
<code>\do@ballastR</code>	1
<code>\do@insidelineLhook</code>	1
<code>\do@insidelineRhook</code>	1
<code>\do@lineL</code>	1
<code>\do@lineLhook</code>	1
<code>\do@lineR</code>	1
<code>\do@lineRhook</code>	1
<code>\do@lockoff</code>	1
<code>\do@lockoffR</code>	1
<code>\do@lockon</code>	1
<code>\do@lockonR</code>	1
<code>\doinsidelineLhook</code>	1
<code>\doinsidelineRhook</code>	1
<code>\dolineLhook</code>	1
<code>\dolineRhook</code>	1
<code>\dump@pstartL@pc</code>	1
<code>\dump@pstartR@pc</code>	1

## E

<code>\edlabel</code>	1
<code>\edtext</code>	1
<code>\eled@sectioningR@out</code>	1
<code>\eledpar@error</code>	1

<code>\eledsection@correcting@skip</code> .....	1
<code>\eledsectmark</code> .....	1, 16
<code>\eledsectnotoc</code> .....	1, 16
<code>\endlock</code> .....	1
<code>\endnumbering</code> .....	1, 11
<code>\endnumberingR</code> .....	1
<code>\endsub</code> .....	1
environments:	
<code>astanza</code> .....	14
<code>Leftside</code> .....	11
<code>pages</code> .....	8
<code>pairs</code> .....	6
<code>Rightside</code> .....	11

## F

<code>\f@x@l@cksR</code> .....	1
<code>\finish@Pages@notes</code> .....	1
<code>\first@linenum@out@Rfalse</code> .....	1
<code>\first@linenum@out@Rtrue</code> .....	1
<code>\firstlinenum</code> .....	1, 12
<code>\firstlinenum*</code> .....	1, 12
<code>\firstlinenumR</code> .....	1, 12
<code>\firstsublinenum</code> .....	1, 12
<code>\firstsublinenum*</code> .....	1, 12
<code>\firstsublinenumR</code> .....	1, 12
<code>\fix@page</code> .....	1
<code>\flag@end</code> .....	1
<code>\flag@start</code> .....	1
<code>\flush@notesR</code> .....	1
<code>\footnoteXmk</code> .....	10
<code>\footnoteXnomk</code> .....	10

## G

<code>\get@nextboxL</code> .....	1
<code>\get@nextboxR</code> .....	1
<code>\getline@numR</code> .....	1
<code>\getlinesfrompagelistL</code> .....	1
<code>\getlinesfrompagelistR</code> .....	1
<code>\getlinesfromparlistL</code> .....	1
<code>\getlinesfromparlistR</code> .....	1
<code>\goalfraction</code> .....	1

## H

<code>\hidenumbering</code> .....	12
-----------------------------------	----

## I

<code>\if@pstarts</code> .....	1
<code>\ifaraw@text</code> .....	1
<code>\iffirst@linenum@out@R</code> .....	1
<code>\ifinstanzaL</code> .....	1

<code>\ifinstanzaR</code> .....	1
<code>\ifl@dpagfull</code> .....	1
<code>\ifl@dpaging</code> .....	1
<code>\ifl@dpairing</code> .....	1
<code>\ifl@dsamepage</code> .....	1
<code>\ifl@dusedbabel</code> .....	1
<code>\ifledRcol</code> .....	1
<code>\ifprevpgnotnumbered</code> .....	1
<code>\ifprint@last@after@pendL</code> .....	1
<code>\ifprint@last@after@pendR</code> .....	1
<code>\ifpst@rtedL</code> .....	1
<code>\ifpst@rtedR</code> .....	1
<code>\ifpstartnumR</code> .....	1
<code>\ifsameparallelpagenunder</code> .....	1
<code>\ifshiftedpstarts</code> .....	1
<code>\ifwidthliketwocolumns</code> .....	1
<code>\ifwrittenlinesL</code> .....	1
<code>\init@series@par</code> .....	1
<code>\initnumbering@sectcountR</code> .....	1
<code>\insert@countR</code> .....	1
<code>\insert@noterule@ledgroup</code> .....	1
<code>\inserthangingsymbolL</code> .....	1
<code>\inserthangingsymbolR</code> .....	1
<code>\insertlines@listR</code> .....	1
<code>\inserts@listR</code> .....	1

## L

<code>\l@d@set</code> .....	1
<code>\l@dbfnote</code> .....	1
<code>\l@dc@maxchunks</code> .....	1
<code>\l@dcalc@maxoftwo</code> .....	1
<code>\l@dcalc@minoftwo</code> .....	1
<code>\l@dcalcnun</code> .....	1
<code>\l@dcheckklang</code> .....	1
<code>\l@dleftbox</code> .....	1
<code>\l@dlinenumR</code> .....	1
<code>\l@dmake@labelsR</code> .....	1
<code>\l@dminpagelines</code> .....	1
<code>\l@dnumpstartsL</code> .....	1
<code>\l@dnumpstartsR</code> .....	1
<code>\l@dpagfullfalse</code> .....	1
<code>\l@dpagfulltrue</code> .....	1
<code>\l@drightbox</code> .....	1
<code>\l@dsamepagefalse</code> .....	1
<code>\l@dsamepagetrue</code> .....	1
<code>\l@dsetupmaxlinecounts</code> .....	1
<code>\l@dsetuprawboxes</code> .....	1
<code>\l@dskipversenumberR</code> .....	1
<code>\l@dusedbabelfalse</code> .....	1
<code>\l@dusedbabeltrue</code> .....	1



<code>\l@duselanguage</code> .....	1
<code>\l@dzeromaxlinecounts</code> .....	1
<code>\l@pscl</code> .....	1
<code>\l@pscR</code> .....	1
<code>\labelref@listR</code> .....	1
<code>\last@page@numR</code> .....	1
<code>\Lcolwidth</code> .....	1, 7, 9
<code>\led@err@BadLeftRightPstarts</code> .....	1
<code>\led@err@Columns@InsideEnv</code> .....	1
<code>\led@err@LeftOnRightPage</code> .....	1
<code>\led@err@Leftside@PreviousNotPrinted</code> .....	1
<code>\led@err@Pages@InsideEnv</code> .....	1
<code>\led@err@RightOnLeftPage</code> .....	1
<code>\led@err@Rightside@PreviousNotPrinted</code> .....	1
<code>\led@err@TooManyPstarts</code> .....	1
<code>\led@error@fail@patch@mempnum</code> .....	1
<code>\led@error@fail@patch@outputpage</code> .....	1
<code>\led@error@fail@patch@pagenumbering</code> .....	1
<code>\led@error@fail@patch@thepage</code> .....	1
<code>\led@nopbnumR</code> .....	1
<code>\led@nopbR</code> .....	1
<code>\led@pbnumR</code> .....	1
<code>\led@pbR</code> .....	1
<code>\lednopbnum</code> .....	1
<code>\lednopbnumR</code> .....	1
<code>\ledpbnumR</code> .....	1
<code>\ledpbR</code> .....	1
<code>\ledstrutL</code> .....	1
<code>\ledstrutR</code> .....	1
<code>\ledthegoal</code> .....	1
<code>\leftlinenumR</code> .....	1
<code>\leftpstartnumL</code> .....	1
<code>\leftpstartnumR</code> .....	1
<code>Leftside (environment)</code> .....	11
<code>\Leftsidehook</code> .....	1
<code>\Leftsidehookend</code> .....	1
<code>\line@list@stuffR</code> .....	1
<code>\line@listR</code> .....	1
<code>\line@marginR</code> .....	1
<code>\line@numR</code> .....	1
<code>\lineation*</code> .....	1, 12
<code>\lineationR</code> .....	1, 12
<code>\linenum@outR</code> .....	1
<code>\linenumberstyle*</code> .....	1, 12
<code>\linenumberstyleR</code> .....	1, 12
<code>\linenumincrement</code> .....	1, 12
<code>\linenumincrement*</code> .....	1, 12
<code>\linenumincrementR</code> .....	1, 12
<code>\linenummargin</code> .....	1
<code>\linenumrepR</code> .....	1

<code>\linesinpar@listL</code> .....	1
<code>\linesinpar@listR</code> .....	1
<code>\list@clearing@regR</code> .....	1
<code>\list@pstartL@pc</code> .....	1
<code>\list@pstartR@pc</code> .....	1
<code>\lock@off</code> .....	1

**M**

<code>\maxchunks</code> .....	1, 6
<code>\maxlinesinpar@list</code> .....	1
<code>\memorydump</code> .....	11
<code>\memorydumpL</code> .....	1
<code>\memorydumpR</code> .....	1

**N**

<code>\n@num</code> .....	1
<code>\namebox</code> .....	1
<code>\new@lineL</code> .....	1
<code>\new@lineR</code> .....	1
<code>\newnamebox</code> .....	1
<code>\newnamecount</code> .....	1
<code>\newseries@par</code> .....	1
<code>\normalbfnoteX</code> .....	1
<code>\notesXwidthliketwocolumns</code> .....	8
<code>\num@linesR</code> .....	1
<code>\numberpstartfalse</code> .....	12
<code>\numberpstarttrue</code> .....	12
<code>\numpagelinesL</code> .....	1
<code>\numpagelinesR</code> .....	1

**O**

<code>\one@lineR</code> .....	1
<code>\onlysideX</code> .....	10

**P**

<code>\page@action</code> .....	1
<code>\page@numR</code> .....	1
<code>\Pages</code> .....	1, 8
<code>pages (environment)</code> .....	8
<code>pairs (environment)</code> .....	6
<code>\par@lineR</code> .....	1
<code>\par@patch@pagenumbering</code> .....	1
<code>\par@patch@thepage</code> .....	1
<code>\parledgroup@</code> .....	1
<code>\parledgroup@beforenotes@save</code> .....	1
<code>\parledgroup@beforenotesL</code> .....	1
<code>\parledgroup@beforenotesR</code> .....	1
<code>\parledgroup@correction@notespacing</code> .....	1
<code>\parledgroup@correction@notespacing@final</code> .....	1
<code>\parledgroup@correction@notespacing@init</code> .....	1

<code>\parledgroup@notes@startL</code> .....	1
<code>\parledgroup@notes@startR</code> .....	1
<code>\parledgroup@notespacing@correction</code> .....	1
<code>\parledgroup@notespacing@set@correction</code> .....	1
<code>\parledgroupseries@</code> .....	1
<code>\parledgrouptype@</code> .....	1
<code>\pausenumberingR</code> .....	1
<code>\pend</code> .....	13
<code>\pendL</code> .....	1
<code>\pendR</code> .....	1
<code>\prev@nopbR</code> .....	1
<code>\prev@pbR</code> .....	1
<code>\prevpgstyle</code> .....	1
<code>\print@columnseparator</code> .....	1
<code>\print@eledsectionL</code> .....	1
<code>\print@eledsectionR</code> .....	1
<code>\print@lineL</code> .....	1
<code>\print@lineR</code> .....	1
<code>\print@notesX@forpages</code> .....	1
<code>\print@Xnotes@forpages</code> .....	1
<code>\printlinesR</code> .....	1
<code>\pstart</code> .....	13
<code>\pstartL</code> .....	1
<code>\pstartR</code> .....	1

## R

<code>\Rcolwidth</code> .....	1, 7, 9
<code>\read@linelist</code> .....	1
<code>\restore@pstartL@pc</code> .....	1
<code>\restore@pstartR@pc</code> .....	1
<code>\resumenumberingR</code> .....	1
<code>\rightlinenumR</code> .....	1
<code>\rightpstartnumL</code> .....	1
<code>\rightpstartnumR</code> .....	1
Rightside (environment) .....	11
<code>\Rightsidehook</code> .....	1
<code>\Rightsidehookend</code> .....	1
<code>\Rlineflag</code> .....	1

## S

<code>\section@numR</code> .....	1
<code>\selectlanguage</code> .....	1
<code>\set@line</code> .....	1
<code>\set@line@action</code> .....	1
<code>\setgoalfraction</code> .....	1, 9
<code>\sethangingsymbol</code> .....	15
<code>\setline</code> .....	1
<code>\setlinenum</code> .....	1
<code>\setnamebox</code> .....	1
<code>\setnotepositionliketwocolumns@C</code> .....	1

<code>\setnotepositionliketwocolumns@L</code> .....	1
<code>\setnotepositionliketwocolumns@R</code> .....	1
<code>\setpositionliketwocolumns@C</code> .....	1
<code>\setpositionliketwocolumns@L</code> .....	1
<code>\setpositionliketwocolumns@R</code> .....	1
<code>\setRlineflag</code> .....	13
<code>\setwidthliketwocolumns@C</code> .....	1
<code>\setwidthliketwocolumns@L</code> .....	1
<code>\setwidthliketwocolumns@R</code> .....	1
<code>\sidenote@marginR</code> .....	1
<code>\sidenotemargin*</code> .....	1
<code>\skip@lockoff</code> .....	1
<code>\skipnumbering</code> .....	1, 12
<code>\startlock</code> .....	1
<code>\startsub</code> .....	1
<code>\sub@action</code> .....	1
<code>\subline@numR</code> .....	1
<code>\sublinenumberstyle*</code> .....	1, 12
<code>\sublinenumberstyleR</code> .....	1, 12
<code>\sublinenumincrement</code> .....	1, 12
<code>\sublinenumincrement*</code> .....	1, 12
<code>\sublinenumincrementR</code> .....	1, 12
<code>\sublinenumrepr</code> .....	1

## T

<code>\theledlanguageL</code> .....	1
<code>\theledlanguageR</code> .....	1
<code>\thepar@page</code> .....	1
<code>\thepstartL</code> .....	12
<code>\thepstartR</code> .....	12
<code>\thestanzaL</code> .....	1, 15
<code>\thestanzaR</code> .....	1, 15

## U

<code>\unhnamebox</code> .....	1
<code>\unvnamebox</code> .....	1
<code>\usernamecount</code> .....	1

## W

<code>\widthliketwocolumns</code> .....	8
---	---

## X

<code>\Xnoteswidthliketwocolumns</code> .....	8
<code>\Xonlyside</code> .....	10

## Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	70
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	76
\Pages: Added \l@duselanguage to \Pages	86
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	51
Added hooks into Leftside environment	42
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match new eledmac	55
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	53
\do@lineL: Added \do@lineLhook to \do@lineL	48
Simplified \do@lineL by using macros for some common code	48
\do@lineR: Changed \do@lineR similarly to \do@lineL	51
\flag@end: Removed extraneous spaces from \flag@end	36
\ifledRcol: Moved \ifl@dpairing to eledmac	19
\ifpst@rtedR: Moved \ifpst@rtedL to eledmac	21
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	28
\l@dnumpstartsR: Moved \l@dnumpstartsL to eledmac	73
\ledstrutR: Added \ledtrutL and \ledstrutR	90
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX	67
\Pages: Added \ledstrutL to \Pages	86
Added \ledstrutR to \Pages	87
\printlinesR: Simplified \printlinesR by using \setprintlines	61
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	42
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	27
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Do not just set \line@marginR in \linenummargin	26
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	88
v0.3.c.	
General: Compatibilty with Polyglossia	1
v0.4.0.	
General: No more ledparpatch. All patches are now in the main file.	1
v0.5.0.	
General: Corrections about \section and other titles in numbered sections	1
v0.6.0.	
General: Be able to us \chapter in parallel pages.	1
v0.7.0.	
General: Option ‘shiftedverses’ which make there is no blank between two parallel verses with unequal length.	1

v0.8.0.	General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande <code>\hangingsymbol</code> to define the character. . . . .	1
v0.9.0.	General: Possibility to number <code>\pstart</code> . . . . .	12
	Possibility to number the <code>pstart</code> with the commands <code>\numberpstarttrue</code> . . . . .	1
	<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code> . . . . .	19
v0.9.1.	General: The numbering of the <code>pstarts</code> restarts on each <code>\beginnumbering</code> . . . . .	1
v0.9.2.	General: Debug : with <code>\Columns</code> , the hanging indentation now runs on the left columns and the hanging symbol is shown only when <code>\stanza</code> is used. . . . .	1
v0.9.3.	General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with <code>memoir</code> class. . . . .	1
v0.10.0.	General: <code>\edlabel</code> commands on the right side are now correctly indicated. . . . .	1
	<code>\edlabel</code> commands which start a paragraph are now put in the right place. . . . .	1
v0.11.0.	General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by <code>pstart</code> (like in <code>eledmac</code> 0.15). . . . .	48
	Lineation can be by <code>pstart</code> (like in <code>eledmac</code> 0.15). . . . .	24
	New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . .	68
	<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac</code> 0.15). . . . .	55
	<code>\Columns</code> : Line numbering by <code>pstart</code> . . . . .	77
	<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in <code>eledmac</code> 0.15). . . . .	95
	<code>Pstart</code> number can be printed in side . . . . .	96
	<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	68
v0.12.0.	General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . . . .	68
v1.0.0.	General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code> . . . . .	1
	Debug in lineation by <code>pstart</code> . . . . .	24
v1.0.1.	General: Correction on <code>\numberonlyfirstinline</code> with lineation by <code>pstart</code> or by page. .	1
v1.1.0.	General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code> . . . . .	1
	<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from <code>eledmac</code> ). . . . .	43
v1.1.1.	<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1. . . . .	43
v1.1.2.	<code>\affixside@noteR</code> : Remove spurious space between line number and line content . .	66
v1.2.0.	General: Support for <code>\led&lt;section&gt;</code> commands in parallel texts. . . . .	1
v1.2.1.	<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once.	23

v1.3.0.	
\edtext: Manage RTL language. . . . .	37
v1.3.1.	
\l@dbfnote: Compatibility of standard footnotes with eledmac when these footnotes contain any commands. . . . .	67
v1.3.2.	
General: Debug with some classes. . . . .	1
v1.3.3.	
General: Debugging the left notes of the right column. . . . .	66
\l@dbfnote: Spurious space with footnote in right column. . . . .	67
v1.3.4.	
General: Allow use of commands in sidenotes, as introduced by eledmac 1.0. . . . .	66
v1.3.5.	
\normalbfnoteX: Allows one to redefine \thefootnoteX with alph when some packages are loaded. . . . .	67
v1.4.0.	
General: Added \do@insidelineLhook and \do@insidelineRhook . . . . .	51
v1.4.1.	
General: Enable the use of stanzaindentsrepetition within astanza environment. . . . .	68
\normalbfnoteX: Fix bug with normal familiar footnotes when mixing RTL and LTR text. . . . .	67
v1.4.3.	
General: Corrects a false hanging verse when a verse is exactly the length of a line. . . . .	1
\inserthangingsymbolR: Hanging verse is no longer automatically flush right. . . . .	68
\pendL: Spurious spaces in \pendL. . . . .	46
\pendR: Spurious spaces in \pendR. . . . .	47
\pstartR: Spurious spaces in \pstartL and \pstartR. . . . .	43
v1.5.0.	
General: Add, as in eledmac, features to manage page breaks. . . . .	1
\sublinenumincrement*: Add starred version of \firstlinenum, \linenumincrement, \firstsublinenum, \sublinenumincrement to change both Left and Rightside. . . . .	26
v1.6.0.	
General: Add tool and documentation for parallel ledgroups . . . . .	15
v1.7.0.	
General: Add, as in eledmac, features to make crossrefs with pstart numbers. . . . .	1
v1.8.0.	
General: \beginnumbering is defined only on eledmac, not on eledpar. . . . .	22
\l@dlsnote, \l@drsnote and \l@dcsnote defined only one time, in eledmac. . . . .	66
Add \beforecolumnseparator and \aftercolumnseparator. . . . .	8
Add \columnspostion. . . . .	7
Add, as in eledmac, new system of sectioning commands. . . . .	1
Add, as in eledmac, option to insert something after \pends / verses. . . . .	1
Add, as in eledmac, option to insert something between \pstarts / verse. . . . .	1
Change \do@lineR and \do@lineR to allow new sectioning commands. . . . .	48
Compatibility with musixtex. . . . .	1
Debug eledmac sectioning command after using \resumenumbering. . . . .	1
New sectioning commands, as in eledmac. . . . .	16
Suppress \ifl@dsamelang which did not work and was not logical, because both columns could have the same language but not the main language of the document. . . . .	71
\Columns: Modify \Columns to enable to add section's title. . . . .	75

Suppress <code>\l@dchecklang</code> from <code>\Columns</code> . . . . .	76
<code>\l@dchecklang</code> : Suppress <code>\l@dchecklang</code> which did not work and was not logical, because both columns could have the same language but not the main language of the document. . . . .	71
<code>\Pages</code> : Modify <code>\Pages</code> to enable to add section's title. . . . .	83
<code>\pendL</code> : As in <code>eledmac</code> , <code>\pendL</code> can have an optional argument. . . . .	46
<code>\pendR</code> : As in <code>eledmac</code> , <code>\pendR</code> can have an optional argument. . . . .	47
<code>\print@columnseparator</code> : Move some code of <code>\Columns</code> to <code>\print@columnseparator</code> . . . . .	78
<code>\pstartR</code> : As in <code>eledmac</code> , <code>\pendL</code> and <code>\pendR</code> can have an optional argument. . . . .	43
<code>\sidenotemargin*</code> : <code>\sidenotemargin</code> is now directly defined in <code>eledmac</code> to be able to manage <code>eledpar</code> . . . . .	66
Add <code>\sidenotemargin*</code> . . . . .	66
<code>\theledlanguageR</code> : Correct left/right language setting with <code>polyglossia</code> . . . . .	72
v1.8.1.	
<code>\do@lineL</code> : Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?). . . . .	48
<code>\do@lineR</code> : Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?). . . . .	51
v1.8.2.	
General: Debug <code>\eledxxx</code> with some paper sizes . . . . .	1
Debug left and side note (bugs added by 1.8.0) . . . . .	1
<code>\eledpar@error</code> : Errors specific to <code>eledpar</code> send to <code>eledpar</code> handbook . . . . .	20
<code>\flag@end</code> : <code>\flag@start</code> and <code>\flag@end</code> are now defined only one time for <code>eledmac</code> and <code>eledpar</code> . . . . .	36
<code>\lineation*</code> : Add <code>\lineation*</code> . . . . .	25
v1.8.3.	
General: Add <code>\noeledxxx</code> , as in <code>eledmac</code> . . . . .	1
<code>\doinsidelineRhook</code> : Added <code>\dolineLhook</code> , <code>\dolineRhook</code> , <code>\doinsidelineLhook</code> and <code>\doinsidelineRhook</code> . . . . .	50
<code>\Pages</code> : Debug blank pages when using optional argument in the last <code>\pend</code> . . . . .	83
<code>\resumenumberingR</code> : Debug <code>\resumenumberingR</code> . . . . .	23
v1.9.0.	
General: Add <code>\AtBeginPairs</code> macro. . . . .	7
Compatibility with <code>\Xnoteswidthliketwocolumns</code> and <code>\notesXwidthliketwocolumns</code> . . . . .	1
<code>\ifwidthliketwocolumns</code> : Added <code>widthliketwocolumns</code> option . . . . .	19
<code>\theledlanguageR</code> : Debug left/right language switching with <code>polyglossia</code> . Do not write in .aux file when setting left/right lines. . . . .	72
v1.9.1.	
<code>\ifledRcol</code> : Moved <code>\ifl@dpaging</code> to <code>eledmac</code> . . . . .	19
v1.10.0.	
General: Compatibility with <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> . . . . .	1
Restore critical notes in <code>\eledsection</code> in parallel columns (this bug was added in 1.8.2). . . . .	1
<code>\Pages</code> : Debug wrong pages splitting when no optional argument is used in last <code>\pend</code> (bug was added in v.1.8.3). . . . .	83
Debug wrong parallel pages synchronization when an <code>\edtext</code> falls across two pages. 83	
v1.10.1.	
<code>\line@list@stuffR</code> : Revert modification of 1.4.2, which makes bugs with numbering. Leave vertical mode to solve spurious space before <code>minipage</code> . . . . .	36



v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles. . . . .	1
\edtext: \critext and \edtext are now defined only in eledmac. . . . .	37
v1.12.0.	
General: Compatibility with Lua $\TeX$ RTL languages. . . . .	1
\Columns: Add \l@dprintingcolumnstrue . . . . .	75
\edlabel: \edlabel and \edindex works now with hyperref when using eledpar. . .	66
\edlabel is now defined only one time for both eledmac and eledpar . . . . .	66
\Pages: Add \l@dprintingpagestrue . . . . .	83
\print@eledsectionL: Compatibility with Lua $\TeX$ RTL languages. . . . .	50
\print@eledsectionR: Compatibility with Lua $\TeX$ RTL languages. . . . .	52
\print@lineL: Compatibility with Lua $\TeX$ RTL languages. . . . .	49
v1.12.1.	
\print@eledsectionL: Fixes bug with Lua $\TeX$ RTL \eledsection. . . . .	50
v1.13.0.	
General: Enable the use of optional argument of & in astanza environment. . . . .	68
Fix bug in shiftedpstarts when size difference between pstarts is very important. . . .	1
With parallel pages, long notes can now flow from the Left to the right side and from the Right to the left side. . . . .	1
\clearl@dtrightpage: Use \newpage instead of \clearpage. . . . .	90
\ifledRcol: Remove false boolean settings which are not needed. . . . .	19
\Pages: Prevent false overfull hboxes when using \Pages outside of pages environment.	84
When using shiftedpstarts option, a \l@dleftbox with a null height will advance the \pagetotal in any case. . . . .	83
v1.13.1.	
\correct@footinsX@box: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages. . . . .	61
Correct \correct@footinsX@box and \correct@Xfootins@box . . . . .	61
\Pages: Prevent false empty page after \Pages (bug added in 1.13.0) . . . . .	83
v1.14.0.	
General: Fix bug with line number position when using \eledsection and similar com- mands for RTL texts with Lua $\TeX$ . . . . .	1
The \newifs are not followed by boolean values set to false, because it is the $\TeX$ default setting. . . . .	1
v1.15.0.	
General: Add \AtEveryPstartCall. . . . .	1
Add sameparallepagenumber option. . . . .	9
Fix vertical spurious space before right \eledchapter (bug added in v1.13.0). . . . .	1
Prevent vertical space when using \AtEveryPstart or \AtEveryPend with a com- mand which prints nothing . . . . .	1
\do@actions@nextR: Add action 1008 and 1009 . . . . .	53
\inserthangingsymbolR: Prevent more efficiently the column separator from shifting when a verse is hanging . . . . .	68
\lineationR: As \lineation, \lineationR automatically set the \pstartinfootnote. . . . . .	25
\n@num: \n@num defined only one time for both Eledmac and Eledpar. . . . .	33
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar . . . . .	37

v1.16.0.	
General: Error message when calling <code>\Pages</code> inside ‘pages’ environment and <code>\Columns</code> inside ‘pairs’ environment. . . . .	1
Error message when starting a Leftside/a Rightside while the previous one has not been yet typeset. . . . .	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code> . . . . .	1
Fix bug with <code>nofamiliar / nocritical</code> option of <code>eledmac</code> . . . . .	1
New package option <code>sameparallelpagenumber</code> to have the same page number for both left and right side. . . . .	1
<code>\newseries@par</code> : Fix bug with <code>\onlysideX</code> . . . . .	37
v1.16.1.	
General: Write information about line-list file version in the correct file. . . . .	1
v1.16.2.	
General: Fix bug when adding empty lines before a <code>\pend</code> in combination with some specific penalties setting. . . . .	1
v1.17.0.	
General: Add compatibility of optional argument of <code>\pstart/\pend</code> and <code>\AtEveryPstart/\AtEveryPend</code> with two columns mode. . . . .	1
v1.21.0.	
General: Add <code>\hidenumbering</code> . . . . .	12
v2.0.0.	
<code>\@adv</code> : <code>\@adv</code> defined only in <code>reledmac</code> . . . . .	31
<code>\@lab</code> : <code>\@lab</code> defined only in <code>eledmac</code> . . . . .	66
<code>\@ref@regR</code> : <code>\@ref</code> defined only in <code>reledmac</code> , code specific to right side moved in <code>\ref@regR</code> . . . . .	33
<code>\@set</code> : <code>\@set</code> defined only in <code>reledmac</code> . . . . .	31
General: <code>\@nl</code> is now defined only in <code>reledmac</code> . . . . .	30
<code>\ifbypage@</code> and <code>\ifbypstart@R</code> defined in <code>eledmac</code> . . . . .	24
Fix some bugs with ‘ <code>sameparallelpagenumber</code> ’ option. . . . .	1
Many code refactored and moved to <code>reledmac</code> . . . . .	1
Package’s name becomes <code>reledpar</code> . . . . .	1
Totally new implementation of ‘ <code>sameparallelpagenumber</code> ’ option. . . . .	1
<code>\advanceline</code> : <code>\advanceline</code> defined only in <code>reledmac</code> . . . . .	36
<code>\bbl@set@language</code> : Patch <code>\bbl@set@language</code> instead of redefining it . . . . .	71
<code>\do@lockonR</code> : <code>\do@lockon</code> defined only in <code>reledmac</code> . . . . .	32
<code>\endlock</code> : <code>\startlock</code> and <code>\endlock</code> defined only in <code>reledmac</code> . . . . .	37
<code>\endsub</code> : <code>\startsub</code> and <code>\endsub</code> defined only in <code>reledmac</code> . . . . .	36
<code>\fix@page</code> : <code>\fix@page</code> is defined only once in <code>reledmac</code> . . . . .	31
<code>chapterinpages</code> : Deleting the old system of managing parallel chapter, keep only the new one with <code>\patchcmd</code> . . . . .	41
<code>\l@d@set</code> : <code>\l@d@set</code> defined only in <code>reledmac</code> . . . . .	31
<code>\l@dbfnote</code> : <code>\l@dbfnote</code> defined only in <code>reledmac</code> . . . . .	67
<code>\line@marginR</code> : <code>\linenummargin</code> now defined only once time in <code>reledmac</code> . . . . .	26
<code>\normalbfnoteX</code> : <code>\normalbfnoteX</code> defined only in <code>reledmac</code> . . . . .	67
<code>\page@action</code> : <code>\page@action</code> defined only in <code>reledmac</code> . . . . .	31
<code>\read@linelist</code> : <code>\read@linelist</code> is defined only once time in <code>\reledmac</code> . . . . .	30
<code>\set@line</code> : <code>\set@line</code> defined only in <code>reledmac</code> . . . . .	37
<code>\set@line@action</code> : <code>\set@line@action</code> defined only in <code>reledmac</code> . . . . .	31
<code>\setline</code> : <code>\setline</code> defined only in <code>reledmac</code> . . . . .	36
<code>\setlinenum</code> : <code>\setlinenum</code> defined only in <code>reledmac</code> . . . . .	37

<code>\skip@lockoff</code> : <code>\do@lockoff</code> defined only in <code>reledmac</code> . . . . .	32
<code>\sub@action</code> : <code>\sub@action</code> defined only in <code>reledmac</code> . . . . .	31
<code>\sublinenumincrement*</code> : <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\firstsublinenum</code> , <code>\sublinenumincrement</code> are now defined only in <code>reledmac</code> . . . . .	26
<code>\theledlanguageR</code> : Patch <code>\otherlanguage</code> instead of redefining it. . . . .	72
v2.1.0.	
General: Fix bug when using <code>\eledsection</code> and related on right pages when page width is short. . . . .	1
Fix bug when using <code>\pagenumbering</code> with <code>memoir</code> (bug added in v2.0.0). . . . .	1
Fix bug with <code>\setparledgroupnotspacing</code> with the <code>shiftedpstarts</code> option. . . . .	1
Fix incompatibility between optional argument of <code>\pstart</code> and <code>\numberpstarttrue</code>	1
Options to custom empty right page before <code>\Pages</code> . . . . .	1
v2.2.0.	
General: <code>astanza</code> environment can take an optional argument, which will be the optional argument of <code>\pstart</code> started by this environment. . . . .	1
New tools to number stanza . . . . .	1
v2.2.1.	
General: Fix bug with optional argument of last left <code>\pend</code> . . . . .	1
v2.3.0.	
General: Change some internal codes in order to provide compatibility with $\LaTeX$ release of october 2015 . . . . .	1
Fix bug with title number in parallel columns . . . . .	1
New line setting command suffixed by R to set only the right side. . . . .	1
<code>\Pages</code> : Fix bug when calling <code>\Columns</code> after a <code>\Pages</code> (bug added in v1.13.0). . . . .	84