

# 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](https://github.com) to access my page ([maieul/ledmac](https://github.com/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 Options</b>	<b>6</b>
2.1 Synchronization’s options . . . . .	6
2.2 Other options . . . . .	6
<b>3 General</b>	<b>6</b>

---

\*This file (`reledpar.dtx`) has version number v2.5.1, last revised 2015/10/19.

<sup>†</sup>`maieul at maieul dot net`

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

<b>4 Parallel columns</b>	<b>8</b>
4.1 Basic use . . . . .	8
4.2 Setting . . . . .	8
4.2.1 Column's width . . . . .	8
4.2.2 Column's separator . . . . .	8
4.2.3 Column's positions . . . . .	9
4.2.4 Mixing two columns and one column texts . . . . .	9
<b>5 Facing pages</b>	<b>9</b>
5.1 Basic usage . . . . .	9
5.2 Setting . . . . .	10
5.2.1 Text width . . . . .	10
5.2.2 Way of synchronizing . . . . .	10
5.2.3 Page number . . . . .	12
5.2.4 Page breaking . . . . .	12
5.2.5 Right page before \Pages . . . . .	12
5.2.6 Notes about \mainmatter . . . . .	12
5.3 Critical and familiar footnotes . . . . .	12
5.3.1 Notes height setting . . . . .	13
5.3.2 Notes for one side only . . . . .	13
5.3.3 Familiar notes called in the right side, but to be printed in the left side	13
<b>6 Left and right texts</b>	<b>14</b>
6.1 Environments . . . . .	14
6.2 Numbering text lines and paragraphs . . . . .	14
6.3 Line numbering scheme . . . . .	15
6.4 Lineation system . . . . .	15
6.5 Chunks . . . . .	16
6.6 \AtEveryPstart and \AtEveryPstartCall . . . . .	16
6.7 Language setting . . . . .	17
<b>7 Verse</b>	<b>17</b>
<b>8 Side notes</b>	<b>18</b>
<b>9 Parallel ledgroups</b>	<b>18</b>
9.1 General . . . . .	18
9.2 Parallel ledgroups and setspace package . . . . .	19
<b>10 Sectioning commands</b>	<b>19</b>
<b>11 Notes about page number</b>	<b>19</b>
<b>I Implementation overview</b>	<b>20</b>

<b>II Preliminaries</b>	<b>20</b>
II.1 Package's meta-data . . . . .	20
II.2 Package's requirement . . . . .	20
II.3 Package's options . . . . .	20
II.4 Package's options . . . . .	21
II.4.1 Synchronization's options . . . . .	21
II.4.2 Other options . . . . .	22
II.5 Determining side and category of parallel processing . . . . .	22
II.6 Text's width . . . . .	23
II.7 Messages . . . . .	23
<b>III Sectioning commands</b>	<b>25</b>
<b>IV Line counting</b>	<b>28</b>
IV.1 Setting lineation reset . . . . .	28
IV.2 Setting line number margin . . . . .	29
IV.3 Setting lineation start and step . . . . .	30
IV.4 Setting line flag . . . . .	31
IV.5 Setting line number style . . . . .	31
IV.6 Print marginal line number . . . . .	32
IV.7 Line-number counters and lists . . . . .	32
IV.7.1 Correspond to those in <code>reledmac</code> for regular or left text . . . . .	32
IV.7.2 Specific to <code>reledpar</code> . . . . .	33
IV.8 Reading the line-list file . . . . .	33
IV.9 Commands within the line-list file . . . . .	34
IV.10 Writing to the line-list file . . . . .	40
<b>V Marking text for notes</b>	<b>41</b>
V.1 Specific hooks and commands for notes . . . . .	42
V.1.1 Notes to be printed on one side only . . . . .	42
V.1.2 Familiar footnotes without marks . . . . .	42
V.1.3 Create hooks . . . . .	44
V.1.4 Init standards series (A,B,C,D,E,Z) . . . . .	44
<b>VI Pstart numbers dumping and restoration</b>	<b>44</b>
<b>VII Parallel environments</b>	<b>45</b>
<b>VIII Paragraph decomposition and reassembly</b>	<b>47</b>
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	48
VIII.2 Processing one line . . . . .	52
VIII.3 Line and page number computation . . . . .	56
VIII.4 Line number printing . . . . .	59
VIII.5 Pstart number printing in side . . . . .	61
VIII.6 Add insertions to the vertical list . . . . .	63
VIII.7 Penalties . . . . .	64
VIII.8 Printing leftover notes . . . . .	65

<b>IX Footnotes</b>	<b>65</b>
IX.1 Line number printing . . . . .	65
IX.2 Footnotes output specific to \Pages . . . . .	66
<b>X Cross referencing</b>	<b>70</b>
<b>XI Side notes</b>	<b>71</b>
<b>XII Familiar footnotes</b>	<b>72</b>
<b>XIII Verse</b>	<b>72</b>
<b>XIV Naming macros</b>	<b>74</b>
<b>XV Fixing babel and polyglossia</b>	<b>75</b>
<b>XVI Counts and boxes for parallel texts</b>	<b>77</b>
<b>XVII Checking text to be processed</b>	<b>79</b>
<b>XVIII Parallel columns</b>	<b>80</b>
<b>XIX Parallel pages</b>	<b>88</b>
XIX.1 Specific counters . . . . .	88
XIX.2 Main macro . . . . .	88
XIX.3 Ensure all notes be printed at the end of parallel pages . . . . .	95
XIX.4 Struts . . . . .	95
XIX.5 Page clearing . . . . .	95
XIX.6 Lines managing . . . . .	97
XIX.7 Page break managing . . . . .	98
XIX.8 Getting boxes content . . . . .	101
<b>XX Page numbering</b>	<b>105</b>
XX.1 Global options . . . . .	105
XX.2 mainmatter option of \Pages . . . . .	107
<b>XXI Sections' titles' commands</b>	<b>108</b>
<b>XXII Page break/no page break, depending on the specific line</b>	<b>108</b>
<b>XXIII Parallel ledgroup</b>	<b>109</b>
<b>XXIV Compatibility with eledmac</b>	<b>113</b>
<b>XXV The End</b>	<b>113</b>

<b>Appendix A Some things to do when changing version</b>	<b>114</b>
Appendix A.1 Migration to <code>eledpar</code> 1.4.3 . . . . .	114
Appendix A.2 Migration from <code>eledpar</code> to <code>reledpar</code> . . . . .	114
Appendix A.2.1 Deprecated options . . . . .	114
Appendix A.2.2 <code>\renewcommandreplaced</code> with <code>command</code> . . . . .	114
Appendix A.2.3 Commands the names of which have changed . . . . .	115
Appendix A.3 Migration to <code>reledpar</code> 2.2.0 . . . . .	115
Appendix A.4 Migration to <code>reledpar</code> 2.3.0 . . . . .	115
Appendix A.5 Migration to <code>reledpar</code> 2.4.0 . . . . .	115
Appendix A.6 Migration to <code>reledpar</code> 2.5.0 . . . . .	115
<b>References</b>	<b>115</b>
<b>Index</b>	<b>116</b>
<b>Change History</b>	<b>134</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 3; 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 Options

The package can be loaded with a number of global options which are listed here. Those options are fully described in the paragraphs devoted to their feature.

### 2.1 Synchronization's options

Please read the paragraph on synchronization's option on 5.2.2 p. 10 to understand better those options.

**shiftedpstarts** prevents white space between paragraphs on facing pages, the white space necessary to sync pages is collected at the bottom of the page instead.

**advancedshiftedpstarts** does the same as **shiftedpstarts**, but the `pstart` shift are not counted to determine when cutting the page. That could help to avoid page with blank lines at the bottom.

**nomaxlines** allows facing pages to have different numbers of lines.

**nosyncpstarts** disables syncing on facing pages. In that case the pages are filled as two streams normal.

### 2.2 Other options

**parledgroup** allows the use of `ledgroup` environment with `reledpar`.<sup>1</sup>

**widthliketwocolumns** set the width of the text printed in a single column to be the same as the width of the text printed in two parallel columns with `reledpar`. This is useful when alternating between normal and parallel typesetting.<sup>2</sup>

**sameparallelpagenumbers** sets page numbers on facing pages to the same value.

**prevpgnotnumbered** enables that the page before facing pages (the one automatically inserted to start parallel pages on a left page) is not counted. This applies only if the page is empty.

## 3 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

---

<sup>1</sup>This option can either be used on `reledmac` or `reledpar`.

<sup>2</sup>This option can either be used on `reledmac` or `reledpar`.

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>3</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 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 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.

---

<sup>3</sup>You can use, anyway, `\numberlinefalse` to disable printing of line numbers.

## 4 Parallel columns

### 4.1 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 6.

`\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.

### 4.2 Setting

#### 4.2.1 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.

#### 4.2.2 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.



### 4.2.3 Column’s positions

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

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

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

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

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

### 4.2.4 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` 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.

## 5 Facing pages

### 5.1 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.

## 5.2 Setting

### 5.2.1 Text width

`\Lcolwidth`   Within the `pages` environment the lengths `\Lcolwidth` and `\Rcolwidth` are the  
`\Rcolwidth`   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.

### 5.2.2 Way of synchronizing<sup>4</sup>

Synchronization of left and right texts in parallel processing requires some ‘numbered’ auxiliary files to be written (namely `.1`, `.1R`, `.2`, `.2R`, and so forth), the content of which may change as long as synchronization is not complete. This usually requires LaTeX to be run several times. Therefore, it is advised to use in conjunction utilities such as `latexmk` to ensure that synchronization is complete.

Numbered paragraphs which are contained between the `\pstart` and `\pend` macros are thereafter called ‘chunks’.

In short, the default setting is designed in such a way that corresponding chunks of text are always kept in synchronization, even at the cost of page padding, as it may result in leaving blank lines between chunks of text. Conversely, using in conjunction `advancedshiftedpstarts` and `nomaxlines` settings ensures that pages are filled with text to full advantage—at the cost of the chunks not being kept in synchronization—and every chunk starts on the facing page of its corresponding chunk.

To understand better how each of the synchronization settings of `reledpar` works, one must first understand how the default setting of `reledpar` synchronizes the left and right chunks.

The aim of the default setting is twofold:

- To ensure that left pages contain what is to be on left sides and that right pages contain what is to be on right sides.
- To ensure that every chunk starts on the page that is facing its corresponding chunk.

As regards the latter, `reledpar` checks that both of the following rules are respected:

---

<sup>4</sup>There is a French version of this article on <http://geekographie.maieul.net/185>.

- The numbers of lines of every pair of chunks must be identical. To keep this rule, `reledpar` may insert some blank lines at the bottom of the chunk that is shorter so that it may eventually have the same number of lines as the one that is longer.
- The main content of two facing pages, apart from critical and familiar footnotes, must have the same numbers of lines, including those that may be blank. Consequently, if one left page contains more notes than the corresponding right page, the bottom of the right page must be left blank.

Each of these rules can be modified by a number of optional synchronization settings in `reledpar`:

1. Regarding the number of lines a pair of chunks may have:
  - (a) 'shiftedpstarts' setting merely moves any added blank lines from the bottom of the chunks to the bottom of the page. It does not allow to have more lines on a given page as it just removes the blank lines between the chunks and does nothing more. To understand better how this work, you may compare the total amounts of lines of text on a given page whether you have activated this setting or not: you will see that both amounts are the same.
  - (b) 'advancedshiftedpstarts' prevents any blank lines from being inserted at the bottom of the chunks, also taking them away from the total amount of lines the page may have. This allows to get more lines on the pages. However, please note that:
    - Blank lines are taken into account as `reledpar` moves from one to the following chunk of text, so that every pair of chunks may always start on the same facing pages.
    - Consequently, blank lines continue to be taken into account in the calculation of the amount of lines a given pair of pages may have. This is why when a longer chunk runs from one page to another the shorter corresponding one also runs across pages, even if this may result in some blank vertical space being left on the first page.
2. As regards the number of lines per page, including blank ones, the `nomaxlines` setting disregards the rule that forces two facing pages to have the same numbers of lines. So it allows to have more text on the pages. Then, by a complex mechanism it is ensured that two corresponding chunks may always start on the same facing pages, provided that `shiftedpstarts` or `advancedshiftedpstarts` settings shall not be activated.

Lastly, one may disregard all of the synchronization rules and content himself with parallel texts typesetting. To achieve this, please use the `nosyncpstarts` setting.

Please note that every change of synchronization setting resets the content of the 'numbered' auxiliary files to make sure that `reledpar` does not try to make the synchronization with wrong calculations.

### 5.2.3 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 `sameparallelpagnumber` 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 11 p. 19.

### 5.2.4 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>}`.

### 5.2.5 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 11 p. 19.

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

### 5.2.6 Notes about `\mainmatter`

If you use `\frontmatter`, do not use `\mainmatter` directly before `\Pages` because it could create spurious empty pages.

Use instead `\pages` with the optional argument `[mainmatter]`. In this case, the content of `\Pages` will start on a left side, without any spurious empty page, and the left pages will be odd (and not event like in normal way), the first one being 1.

## 5.3 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.

### 5.3.1 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>5</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  $\LaTeX$ '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 (6.10.4 p. 38). As an advisable setting:

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

### 5.3.2 Notes for one side only

`\Xonlyside` You may want to typeset notes on one side only (either left or right). Use `\Xonlyside[⟨s⟩]{⟨p⟩}` 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.

### 5.3.3 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 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>5</sup>The same applies to  $\LaTeX$  normal notes. Read <http://tex.stackexchange.com/a/228283/7712> for technical informations.

```

\pend
\endnumbering
\end{Rightside}

```

## 6 Left and right texts

### 6.1 Environments

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

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

### 6.2 Numbering text lines and paragraphs

**\beginnumbering** Each section of numbered text must be preceded by \beginnumbering and followed by  
**\endnumbering** \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 T<sub>E</sub>X'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.

### 6.3 Line numbering scheme

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

### 6.4 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.

The lineation commands which finish by a `R` apply for right text. The lineation commands which are starred apply for both left and right texts. The lineation command which does not finish by a `R` and who are not starred apply for the left side. **However, these commands apply to right side when they are called inside a left environment. However, such features should not be used any more. The recommended practice is to add all setting commands to the preamble.**

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

```

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

```

text. `\sublinenumberstyleR` is the equivalent of `reledmac \sublinenumberstyle` right text. The starred version are for both side.

`\linenummarginR` `\linenummarginR{<margin>}` sets the line margin for right side. `\linenummargin*{<margin>}` sets for both side. `<margin>` can be, as for `reledmac`'s `\linenummargin` one of these values: left, right, inner, outer. 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.

## 6.5 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. 16).

## 6.6 \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`.



## 6.7 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 important to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default. Also, it is the *last* 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.

## 7 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.

`astanza` `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 latter 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 \szi@@@' 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 times. 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. 42) 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.

`\thestanzaR`

## 8 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.

## 9 Parallel ledgroups

### 9.1 General

You can also make parallel ledgroups (see the documentation of `reledmac` about ledgroups, 9 p. 43). 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

```

## 9.2 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.*

## 10 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).

## 11 Notes about page number

If you use `sameparallepagenumber` option (5.2.3 p. 12) or `prevpgnotnumbered` option (5.2.5 p. 12), 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

$\text{\TeX}$  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,  $\text{\TeX}$  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  $\text{\TeX}$  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  $\text{\LaTeX}2\text{\epsilon}$ . 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}[2015/10/19 v2.5.1 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 %
```

## II.4 Package's options

### II.4.1 Synchronization's options

`\@par@this@sync@option` The `\par@sync@option` stores the options of synchronization. It use to ensure these options do not change between two run.

```
10 \def\@par@this@sync@option{%
11 %
```

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.

```
\ifshiftedpstarts12 \newif\ifshiftedpstarts
13 \DeclareOptionX{shiftedpstarts}{%
14   \shiftedpstartstrue%
15   \apptocmd{\@par@this@sync@option}{shifted}{-}{-}%
16 }%
17 %
```

With the option 'advancedshiftedpstarts' 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. Differing to `shiftedpstarts`, the pstart shift are not counted to determine when cutting the page. That could help to avoid page with blank lines at the bottom.

```
\ifshiftedpstarts18 \newif\ifadvancedshiftedpstarts
19 \DeclareOptionX{advancedshiftedpstarts}{%
20   \advancedshiftedpstartstrue%
21   \shiftedpstartstrue%
22   \apptocmd{\@par@this@sync@option}{advancedshifted}{-}{-}%
23 }%
24 %
```

With the option `nomaxlines`, `reledpar` allows facing pages to have not the same number of lines.

```
\ifnomaxlines25 \newif\ifnomaxlines%
26 \DeclareOptionX{nomaxlines}{%
27   \nomaxlinestrue%
28   \apptocmd{\@par@this@sync@option}{nomax}{-}{-}%
29 }%
30 %
```

With the option `nosyncpstarts`, `reledpar` only alternate between left and right side, and does not try to obtain the same number of line in corresponding page.

```

\ifnosyncpstarts 31 \newif\ifnosyncpstarts%
32 \DeclareOptionX{nosyncpstarts}{%
33   \shiftedpstartstrue%
34   \nomaxlinesttrue%
35   \nosyncpstartstrue%
36   \apptocmd{\@par@this@sync@option}{nosync}{-}{-}%
37 }%
38 %

```

#### II.4.2 Other options

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

```

39 \DeclareOptionX{parledgroup}{\parledgrouptrue}
40 %

```

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

```

41 \DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}%
42 %

```

`\ifsameparallelpagenumber` Options related to page numbering

`\ifprevpgnotnumbered`

```

43 \newif\ifsameparallelpagenumber
44 \newif\ifprevpgnotnumbered
45 \DeclareOptionX{sameparallelpagenumber}{\sameparallelpagenumbertrue}
46 \DeclareOptionX{prevpgnotnumbered}{\prevpgnotnumberedtrue}
47 %

```

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

```

48 \DeclareOptionX{prevpgstyle}{\gdef\prevpgstyle{#1}}%
49 %

```

```

50 \ProcessOptionsX%
51 %

```

#### II.5 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 also set TRUE if we are doing parallel pages. `\ifledRcol` is set TRUE if we are doing the right hand text. They are defined in `reledmac`.

## II.6 Text's width

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

```

52 \newdimen\Lcolwidth
53 \Lcolwidth=0.45\textwidth
54 \newdimen\Rcolwidth
55 \Rcolwidth=0.45\textwidth
56 %

```

## II.7 Messages

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

```

\reledpar@error57 \newcommand{\reledpar@error}[2]{\PackageError{reledpar}{#1}{#2}}
58 %

```

```

\reledpar@warning59 \newcommand{\reledpar@warning}[1]{\PackageWarning{reledpar}{#1}}%
60 %

```

```

\led@err@TooManyPstarts61 \newcommand*\led@err@TooManyPstarts{%
62 \reledpar@error{Too many \string\pstart\space without printing.
63 Some text will be lost}{\@ehc}}
64 %

```

```

\led@err@BadLeftRightPstarts65 \newcommand*\led@err@BadLeftRightPstarts}[2]{%
66 \reledpar@error{The numbers of left (#1) and right (#2)
67 \string\pstart s do not match}{\@ehc}}
68 %

```

```

\led@err@LeftOnRightPage69 \newcommand*\led@err@LeftOnRightPage{%
\led@err@RightOnLeftPage70 \reledpar@error{The left page has ended on a right page}{\@ehc}}
71 \newcommand*\led@err@RightOnLeftPage{%
72 \reledpar@error{The right page has ended on a left page}{\@ehc}}
73 %

```

```

\led@err@Leftside@PreviousNotPrinted74 \newcommand*\led@err@Leftside@PreviousNotPrinted{%
\led@err@Rightside@PreviousNotPrinted75 \reledpar@error{You call a new Leftside environment while the previous
one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
76 \newcommand*\led@err@Rightside@PreviousNotPrinted{%
77 \reledpar@error{You call a new Rightside environment while the previous
one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
78 %

```

```

\led@err@Pages@InsideEnv 79 \newcommand*\led@err@Pages@InsideEnv}{%
\led@err@Columns@InsideEnv 80 \reledpar@error{\string\Pages\space must be called *outside* of the `
                             pages` environment}{\@ehc}}
81 \newcommand*\led@err@Columns@InsideEnv}{%
82 \reledpar@error{\string\Columns\space must be called *outside* of the `
                             pairs` environment}{\@ehc}}
83 %

```

```

\led@error@fail@patch@thepage 84 \newcommand{\led@error@fail@patch@thepage}{%
85 \reledpar@error{Fail to patch \string\thepage\space command.}{\@ehc}%
86 }%
87 %

```

```

\led@error@fail@patch@pagenumbering 88 \newcommand{\led@error@fail@patch@pagenumbering}{%
89 \reledpar@error{Fail to patch \string\pagenumbering\space command.}{\@ehc
90 }%
91 }%
92 %

```

```

\led@error@fail@patch@mempnum 92 \newcommand{\led@error@fail@patch@mempnum}{%
93 \reledpar@error{Fail to patch \string@mempnum\space command.}{\@ehc}%
94 }%
95 %

```

```

\led@error@fail@patch@@outputpage 96 \newcommand{\led@error@fail@patch@@outputpage}{%
97 \reledpar@error{Fail to patch \string\outputpage\space command.}{\@ehc}%
98 }%
99 %

```

```

\led@warn@ChangeSyncOption 100 \newcommand*\led@warn@ChangeSyncOption}[1]{%
101 \reledpar@warning{You have changed synchronization's options since last
run. We have not read line-list file #1. Please run LaTeX again.}%
102 }%
103 %

```

```

\led@warn@setting@in@rightside 104 \newcommand{\led@warn@setting@in@rightside}[1]{%
105 \reledpar@warning{You use #1 inside rightside environment.\MessageBreak
106 Such behavior is deprecated.\MessageBreak
107 Use instead #1R or #1* in your preamble}.
108 }
109 %

```



### 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\section@num`, where `nn` is the section number. However, for right side texts the file is called `\jobname\section@numR`. The `\extensionchars` applies to the right side files just as it does to the normal files.

```
110 \newcount\section@numR
111 \section@numR=\z@
112 %
```

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

```
113 \pst@rtedLfalse
114 \newif\ifpst@rtedR
115
116 %
```

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

```
117 \newcommand*{\beginnumberingR}{%
118   \ifnumberingR
119     \led@err@NumberingStarted
120     \endnumberingR
121   \fi
122   \global\l@dnumpststartsR \z@
123   \global\pst@rtedRfalse
124   \global\numberingRtrue
125   \global\advance\section@numR \@ne
126   \global\absline@numR \z@
127   \gdef\normal@page@breakR{}
128   \gdef\l@prev@pbR{}
129   \gdef\l@prev@nopbR{}
130   \global\line@numR \z@
131   \global\@lockR \z@
132   \global\sub@lockR \z@
133   \global\sublines@false
134   \global\let\next@page@numR\relax
135   \global\let\sub@change\relax
136   \message{Section \the\section@numR R }%
137   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
138   \l@dend@stuff
139   \setcounter{pstartR}{1}
140   \begingroup
141   \initnumbering@sectcountR
142   \gdef\eled@sectionsR@{}}%
```

```

143 \if@noeled@sec\else%
144 \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@numR R
145 }{}{}\makeatother%
146 \immediate\openout\eled@sectioningR@out=\jobname.eledsec\the\
147 section@numR R\relax%
148 \fi%
149 }
150 %

```

`\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.

```

149 \def\endnumberingR{%
150 \ifnumberingR
151 \global\numberingRfalse
152 \normal@pars
153 \ifnum\l@dnumpstartsR=0%
154 \led@err@NumberingWithoutPstart%
155 \fi%
156 \ifl@dpairing
157 \global\pst@rtedRfalse
158 \else
159 \ifx\insertlines@listR\empty\else
160 \global\noteschanged@true
161 \fi
162 \ifx\line@listR\empty\else
163 \global\noteschanged@true
164 \fi
165 \fi
166 \ifnoteschanged@
167 \led@mess@NotesChanged
168 \fi
169 \else
170 \led@err@NumberingNotStarted
171 \fi
172 \endgroup
173 \if@noeled@sec\else%
174 \immediate\closeout\eled@sectioningR@out%
175 \fi%
176 }
177 %
178 %

```

`\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  $\TeX$  counter in `\numberingR`.

```

179 \newcounter{chapterR}
180 \newcounter{sectionR}
181 \newcounter{subsectionR}
182 \newcounter{subsubsectionR}
183 \newcommand{\initnumbering@sectcountR}{
184   \let\c@chapter\c@chapterR
185   \let\c@section\c@sectionR
186   \let\c@subsection\c@subsectionR
187   \let\c@subsubsection\c@subsubsectionR
188 }
189 %

```

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

```

\resumenumberingR
190 \newcommand*{\pausenumberingR}{%
191   \endnumberingR\global\numberingRtrue}
192 \newcommand*{\resumenumberingR}{%
193   \ifnumberingR
194     \global\pst@rtedRtrue
195     \global\advance\section@numR \@ne
196     \led@mess@SectionContinued{\the\section@numR R}%
197     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
198     \l@dend@stuff
199     \begingroup%
200     \initnumbering@sectcountR%
201   \else
202     \led@err@numberingShouldHaveStarted
203     \endnumberingR
204     \beginnumberingR
205   \fi}
206
207 %

```

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

```

208 \newcommand*{\memorydumpl}{%
209   \endnumbering
210   \numberingtrue
211   \global\pst@rtedLtrue
212   \global\advance\section@num \@ne
213   \led@mess@SectionContinued{\the\section@num}%
214   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
215   \l@dend@stuff}
216
217 \newcommand*{\memorydumpR}{%
218   \endnumberingR
219   \numberingRtrue
220   \global\pst@rtedRtrue
221   \global\advance\section@numR \@ne

```

```

222 \led@mess@SectionContinued{\the\section@numR R}%
223 \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
224 \l@dend@stuff}
225
226 %

```

## 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`.

```

227 \newcommand*{\lineationR}[1]{\{%
228   \ifnumbering
229     \led@err@LineationInNumbered
230   \else
231     \def\@tempa{#1}\def\@tempb{page}%
232     \ifx\@tempa\@tempb
233       \global\bypage@Rtrue
234       \global\bypstart@Rfalse
235       \unless\ifnocritical@%
236         \Xpstart[] [false]%
237       \fi%
238     \else
239       \def\@tempb{pstart}%
240       \ifx\@tempa\@tempb
241         \global\bypage@Rfalse
242         \global\bypstart@Rtrue
243         \unless\ifnocritical@%
244           \Xpstart%
245         \fi%
246       \else
247         \def\@tempb{section}
248         \ifx\@tempa\@tempb
249           \global\bypage@Rfalse%
250           \global\bypstart@Rfalse%
251           \unless\ifnocritical@%
252             \Xpstart[] [false]%
253           \fi%
254         \else
255           \led@warn@BadLineation
256         \fi%
257       \fi

```

```

258 \fi
259 \fi}}
260 %

```

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

```

261 \WithSuffix\newcommand\lineation*[1]{%
262 \lineation{#1}%
263 \lineationR{#1}%
264 }%
265 %

```

## 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 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.

**\line@marginR**

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`.

```

266 \newcount\line@marginR
267 %

```

By default put right text numbers at the right.

```

268 \line@marginR=\@ne
269
270 %

```

**\linenummarginR** \linenummarginR applies directly for right side, while **\linenummargin\*** applies for both side.

```

271 \newcommand{\linenummarginR}[1]{%
272 \l@getline@margin{#1}%
273 \ifnum\@l@tempcntb>\m@ne%
274 \global\line@marginR=\@l@tempcntb%
275 \fi%
276 }
277 \WithSuffix\newcommand\linenummargin*[1]{%
278 \l@getline@margin{#1}%
279 \ifnum\@l@tempcntb>\m@ne%
280 \global\line@marginR=\@l@tempcntb%
281 \global\line@margin=\@l@tempcntb%
282 \fi%
283 }
284 %

```

### IV.3 Setting lineation start and step

`\c@firstlinenumR`    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.

```
285 \newcounter{firstlinenumR}
286 \setcounter{firstlinenumR}{5}
287 \newcounter{linenumincrementR}
288 \setcounter{linenumincrementR}{5}
289 %
```

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

```
290 \newcounter{firstsublinenumR}
291 \setcounter{firstsublinenumR}{5}
292 \newcounter{sublinenumincrementR}
293 \setcounter{sublinenumincrementR}{5}
294
295 %
```

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

```
296 \firstsublinenum
297 \firstlinenum*
298 \firstsublinenum*
299 \firstlinenum
300 \firstsublinenum*
301 \firstlinenum*
302 \firstsublinenum
303 \firstlinenum*
304 \firstsublinenum
305 \firstlinenum*
306 \firstsublinenum
307 \firstlinenum*
308 \firstsublinenum
309 \firstlinenum*
310 \firstsublinenum
311 \firstlinenum*
312 \firstsublinenum
```

`\firstlinenumR`    And the 'R' suffixed version.

```
313 \firstlinenumR
314 \firstsublinenumR
315 \firstlinenumR
316 \firstsublinenumR
```

```

316 \newcommand\linenumincrementR[1]{%
317   \setcounter{linenumincrementR}{#1}%
318 }
319 \newcommand\firstsublinenumR[1]{%
320   \setcounter{subfirstlinenumR}{#1}%
321 }
322 \newcommand\sublinenumincrementR[1]{%
323   \setcounter{sublinenumincrementR}{#1}%
324 }
325 %

```

## IV.4 Setting line flag

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

```

326 \newcommand{\setRlineflag}[1]{%
327   \gdef\Rlineflag{#1}%
328 }
329 \setRlineflag{R}
330 %

```

## IV.5 Setting line number style

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

```

331 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
332 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
333
334 %

```

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

```

335 \newcommand*{\linenumberstyleR}[1]{%
336   \def\linenumrepR##1{\@nameuse{@#1}{##1}}
337 \newcommand*{\sublinenumberstyleR}[1]{%
338   \def\sublinenumrepR##1{\@nameuse{@#1}{##1}}
339 %

```

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

```

340 \WithSuffix\newcommand\linenumberstyle*[1]{%
341   \linenumberstyle{#1}%
342   \linenumberstyleR{#1}%
343 }%
344
345 \WithSuffix\newcommand\sublinenumberstyle*[1]{%
346   \sublinenumberstyle{#1}%
347   \sublinenumberstyleR{#1}%

```

```

348 }%
349 %
350 %

```

## 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`.

```

351 \newcommand*{\leftlinenumR}{%
352   \l@dlinenumR
353   \kern\linenumsep}
354 \newcommand*{\rightlinenumR}{%
355   \kern\linenumsep
356   \l@dlinenumR}
357 \newcommand*{\l@dlinenumR}{%
358   \numlabfont\linenumrepR{\line@numR}\@Rlineflag%
359   \ifsublines@
360     \ifnum\subline@num>\z@
361       \unskip\fullstop\sublinenumrepR{\subline@numR}%
362     \fi
363   \fi}
364
365 %

```

## 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.

```

366 \newcount\line@numR
367 \newcount\subline@numR
368 \newcount\absline@numR
369
370 %

```

`\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` Here are the commands to create these lists:

`\actionlines@listR`  
`\actions@listR`



```

371 \list@create{\line@listR}
372 \list@create{\insertlines@listR}
373 \list@create{\actionlines@listR}
374 \list@create{\actions@listR}
375
376 %

```

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

```

377 \newcount\page@numR
378
379 %

```

#### IV.7.2 Specific to reledpar

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

```

380 \list@create{\linesinpar@listL}
381 \list@create{\linesinpar@listR}
382 \list@create{\maxlinesinpar@list}
383
384 %

```

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

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

```

385 \newcommand{\list@clearing@regR}{%
386   \list@clear{\line@listR}%
387   \list@clear{\insertlines@listR}%
388   \list@clear{\actionlines@listR}%
389   \list@clear{\actions@listR}%
390   \list@clear{\linesinpar@listR}%
391   \list@clear{\linesonpage@listR}
392 }
393 %

```

`\@par@sync@option` When typesetting parallel pages, `\@par@sync@option` check if we have changed the synchronization's option since the last run. If true, we just not read the numbered file.

```

394 \newcommand{\@par@sync@option}[1]{%
395   \IfStrEq{#1}{\@par@this@sync@option}%
396   {}%
397   {\ifledRcol%
398     \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
399   }%
400   \else%

```

```

400 \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
401 %
402 \fi%
403 \endinput%
404 }%
405 %

```

`\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.

```

406 \newcommand{\@nl@regR}{%
407   \ifx\l@dchset@num\relax \else
408     \advance\absline@numR \@ne
409     \set@line@action
410     \let\l@dchset@num\relax
411     \advance\absline@numR \m@ne
412     \advance\line@numR \m@ne% % do we need this?
413   \fi
414   \advance\absline@numR \@ne
415   \ifx\next@page@numR\relax \else
416     \page@action
417     \let\next@page@numR\relax
418   \fi
419   \ifx\sub@change\relax \else
420     \ifnum\sub@change>\z@
421       \sublines@true
422     \else
423       \sublines@false
424     \fi
425     \sub@action
426     \let\sub@change\relax
427   \fi
428   \ifcase\@lockR
429   \or
430     \@lockR \tw@
431   \or\or

```

```

432 \@lockR \z@
433 \fi
434 \ifcase\sub@lockR
435 \or
436 \sub@lockR \tw@
437 \or\or
438 \sub@lockR \z@
439 \fi
440 \ifsublines@
441 \ifnum\sub@lockR<\tw@
442 \advance\subline@numR \@ne
443 \fi
444 \else
445 \ifnum\@lockR<\tw@
446 \advance\line@numR \@ne \subline@numR \z@
447 \fi
448 \fi}
449
450
451 %

```

`\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`.

```

452 \newcount\last@page@numR
453 \last@page@numR=-10000
454
455 %

```

`\@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`.

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

```

456 \newcount\@lockR
457 \newcount\sub@lockR
458
459 \newcommand*\do@lockonR{%
460   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
461   \ifsublines@
462     \xright@appenditem{-1005}\to\actions@listR
463     \ifnum\sub@lockR=\z@
464       \sub@lockR \@ne
465     \else
466       \ifnum\sub@lockR=\thr@@
467         \sub@lockR \@ne
468       \fi
469     \fi
470   \else
471     \xright@appenditem{-1003}\to\actions@listR
472     \ifnum\@lockR=\z@
473       \@lockR \@ne
474     \else
475       \ifnum\@lockR=\thr@@
476         \@lockR \@ne
477       \fi
478     \fi
479   \fi}
480
481 %

```

`\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`

```

482
483
484 \newcommand*\do@lockoffR{%
485   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
486   \ifsublines@
487     \xright@appenditem{-1006}\to\actions@listR
488     \ifnum\sub@lockR=\tw@
489       \sub@lockR \thr@@
490     \else
491       \sub@lockR \z@
492     \fi
493   \else
494     \xright@appenditem{-1004}\to\actions@listR
495     \ifnum\@lockR=\tw@

```

```

496 \@lockR \thr@@
497 \else
498 \@lockR \z@
499 \fi
500 \fi}
501
502
503 %

```

`\n@num`

`\@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`.

```

504 \newcount\insert@countR
505 %

```

- #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.

```

506 \newcommand*{\@ref@regR}[2]{%
507   \global\advance\@edtext@level by 1%
508   \global\insert@countR=#1\relax
509   \loop\ifnum\insert@countR>\z@
510     \xright@appenditem{\the\absline@numR}\to\insertlines@listR
511     \global\advance\insert@countR \m@ne
512   \repeat
513 %

```

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.

```

514 \begingroup
515 \let\@ref=\dummy@ref
516 \let\@lopR\@gobble
517 \let\page@action=\relax
518 \let\sub@action=\relax

```

```

519 \let\set@line@action=\relax
520 \let\@lab=\relax
521 \let\@lemma=\relax
522 \let\@sw\@gobblethree%
523 #2
524 \global\endpage@num=\page@numR
525 \global\endline@num=\line@numR
526 \global\endsubline@num=\subline@numR
527 \endgroup
528 %

```

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

```

529 \xright@appenditem%
530 {\the\page@numR|\the\line@numR|}%
531 \ifsublines@ \the\subline@numR \else 0\fi}%
532 \the\endpage@num|\the\endline@num|}%
533 \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
534 %

```

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

```

535 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
@edtext@level\endcsname}%
536 %

```

Declare and init boolean for lemma in this level.

```

537 \providebool{lemmacommand@\the\@edtext@level}%
538 \boolfalse{lemmacommand@\the\@edtext@level}%
539 %

```

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

```

540 #2
541 % Now, we store the list of \protect\cs{@sw} of this current \protect\cs{
edtext} as an element of
542 % the global list of list of \protect\cs{@sw} for a \protect\cs{edtext}
depth.
543 % \begin{macrocode}
544 \ifnum\@edtext@level>0%
545 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtextR@\the\@edtext@level\endcsname}}%
546 \ifcsundef{sw@list@edtextR@\the\@edtext@level}{\
create@this@edtext@level}%
547 \letcs{\@tmp}{sw@list@edtextR@\the\@edtext@level}%
548 \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}%
549 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
550 \global\cslet{sw@list@edtextR@\the\@edtext@level}{\@tmp}%
551 \fi%
552 %

```

Decrease edtext level counter.

```

553 \global\advance\@edtext@level by -1%
554 }
555 %

```

**\@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`.

```

556 \renewcommand*\@pend}[1]{%
557 \ifbypstart@global\line@num=0\fi%
558 \xright@appenditem{#1}\to\linesinpar@listL}
559 \renewcommand*\@pendR}[1]{%
560 \ifbypstart@R\global\line@numR=0\fi
561 \xright@appenditem{#1}\to\linesinpar@listR}
562 %
563 %

```

**\@pstart** **\@pstart** and `cs@pstartR` allows us to know, when using `\nomaxlines` option in which page we should start a pstart, and also how many empty lines we should let before starting this pstart at the beginning of the page

```

564 \newcommand{\@pstart}[3]{%
565 \ifcsdef{minpage@pstart@#1}%
566   {\ifnumgreater{#2}{\csuse{minpage@pstart@#1}}}%
567   {\csnumgdef{minpage@pstart@#1}{#2}}}%
568   {}%
569   }%
570   {\csnumgdef{minpage@pstart@#1}{#2}}
571 \csnumgdef{afterlines@pstart@#1L}{#3}%
572 }%
573
574 \newcommand{\@pstartR}[3]{%
575 \numdef{\@tmp}{#2-1}%Because we have not to know in which page the pstart
576   starts, but in which pair of facing page
577   \ifcsdef{minpage@pstart@#1}%
578     {\ifnumgreater{\@tmp}{\csuse{minpage@pstart@#1}}}%
579     {\csnumgdef{minpage@pstart@#1}{\@tmp}}}%
580     {}%
581     {\csnumgdef{minpage@pstart@#1}{\@tmp}}
582   \csnumgdef{afterlines@pstart@#1R}{#3}%
583 }%
584 %

```

**\@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`.

```

585 \renewcommand*{\@lopL}[1]{%
586   \xright@appenditem{#1}\to\linesonpage@listL}
587 \renewcommand*{\@lopR}[1]{%
588   \xright@appenditem{#1}\to\linesonpage@listR}
589
590 %

```

## 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`.

```

591 \newwrite\linenum@outR
592 %

```

`\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
593 \newif\iffirst@linenum@out@R
594 \first@linenum@out@Rtrue
595 %

```

`\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.

```

596 \newcommand*{\line@list@stuffR}[1]{%
597   \read@linelist{#1}%
598   \iffirst@linenum@out@R
599     \immediate\closeout\linenum@outR
600     \global\first@linenum@out@Rfalse
601     \immediate\openout\linenum@outR=#1
602     \immediate\write\linenum@outR{\string\line@list@version{\
this@line@list@version}}}%
603     \ifl@dpaging%
604       \immediate\write\linenum@outR{\string\@par@sync@option{\
@par@this@sync@option}}}%
605     \fi%
606   \else
607     \if@minipage%
608       \leavevmode%
609     \fi%
610     \closeout\linenum@outR%
611     \openout\linenum@outR=#1%
612   \fi}
613
614 %

```



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

```
615 \newcommand*{\new@lineL}{%
616   \write\linenum@out{\string\@nl[\the\c@page][\thepage]}}
617 %
```

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

```
618 \newcommand*{\new@lineR}{%
619   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}
620 %
```

`\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`.

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

`\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` `\endlock` 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`.

`\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@par621 \newcommand{\newseries@par}[1]{%
622 %
```

### 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.

```
623 \unless\ifnofamiliar@%
624 \csgdef{onlysideX@#1}{}%
625 \global\newbool{keepforsideX@#1}%
626 \fi%
627 \unless\ifnocritical@%
628 \global\newbool{keepforXside@#1}%
629 \csgdef{Xonlyside@#1}{}%
630 \fi%
631 %
```

### 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`.

```
632 \unless\ifnofamiliar@%
633 % So declare the list.
634 % \begin{macrocode}
635 \expandafter\list@create\csname footnote#1@mk\endcsname%
636 %
```

Then, declare the `\footnoteXnomk` command.

```
637 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
638 %
```

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

```
639 \toggletrue{nomk@}%
640 \csuse{footnote#1}{##1}%
641 \togglefalse{nomk@}%
642 %
```

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`.

```

643     \letcs{\@tmp}{footnote#1@mk}%
644     \numdef\@tmpa{\csuse{c@footnote#1}}%
645     \global\xright@appenditem{\@tmpa}\to\@tmp%
646     \global\cslet{footnote#1@mk}{\@tmp}%
647 }%
648 %

```

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

```

649     \expandafter\newcommand\csname footnote#1mk\endcsname{%
650 %

```

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

```

651     \letcs{\@tmp}{footnote#1@mk}%
652     \gl@p\@tmp\to\@tmpa%
653     \global\cslet{footnote#1@mk}{\@tmp}%
654 %

```

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

```

655     \letcs{\old@footnote}{c@footnote#1}%
656     \setcounter{footnote#1}{\@tmpa}%
657 %

```

Define the footnote mark and print it

```

658     \protected@csxdef{\thefnmark#1}{\csuse{thefootnote#1}}%
659     \csuse{\@footnotemark#1}%
660 %

```

Restore previous footnote counter and finally add space.

```

661     \setcounter{footnote#1}{\old@footnote}%
662     \xspace%
663 }%
664 %

```

End of tools for familiar notes without marks

```

665 \fi
666 %

```

End of `\newseries@par`.

```

667 }%
668 %

```

### V.1.3 Create hooks

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

```

669 \unless\ifnocritical@%
670   \newhookcommand@series{Xonlyside}%
671 \fi%
672 \unless\ifnofamiliar@%
673   \newhookcommand@series{onlysideX}%
674 \fi
675
676
677 %

```

### 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`.

```

678 \newcommand{\init@series@par}{%
679   \def\do##1{\newseries@par{##1}}%
680   \dolistloop{\@series}%
681 }%
682 \init@series@par%
683 %

```

## 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@pc84 \list@create{\list@pstartL@pc}%
\list@pstartR@pc85 \list@create{\list@pstartR@pc}%
86 %

```

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

```

\dump@pstartL@pc 87 \def\dump@pstartL@pc{%
\dump@pstartR@pc 88 \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
689 \global\cslet{numberpstart@L\the\l@dnumpstartsL}{\ifnumberpstart}%
690 }%
691
692 \def\dump@pstartR@pc{%
693 \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
694 \global\cslet{numberpstart@R\the\l@dnumpstartsR}{\ifnumberpstart}%
695 }%
696
697 %

```

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

```

\restore@pstartR@pc
698 \def\restore@pstartL@pc{%
699 \ifx\list@pstartL@pc\empty\else%
700 \gl@p\list@pstartL@pc\to\@temp%
701 \global\c@pstartL=\@temp%
702 \fi%
703 }%
704 \def\restore@pstartR@pc{%
705 \ifx\list@pstartR@pc\empty\else%
706 \gl@p\list@pstartR@pc\to\@temp%
707 \global\c@pstartR=\@temp%
708 \fi%
709 }%
710 %

```

## 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.

```

711 \newenvironment{pairs}{%}
712 \l@dpairingtrue
713 \l@dpagingfalse
714 \initnumbering@quote
715 \at@begin@pairs%
716 }{%
717 \l@dpairingfalse
718 }
719
720 %

```

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

```

721 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
722 \def\at@begin@pairs{}%
723
724 %

```

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.

```

725 \newenvironment{pages}{%
726   \l@dpairingtrue
727   \l@dpagingtrue
728   \initnumbering@quote
729   \setlength{\Lcolwidth}{\textwidth}%
730   \setlength{\Rcolwidth}{\textwidth}%
731 }{%
732   \l@dpairingfalse
733   \l@dpagingfalse
734 }
735
736 %

```

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

**ifinstanzaR**

```

737 \newif\ifinstanzaL
738 \newif\ifinstanzaR
739 %

```

**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.

```

740 \newenvironment{Leftside}{%
741   \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
742   \led@err@Leftside@PreviousNotPrinted%
743   \fi%
744   \ledRcolfalse
745   \setcounter{pstartL}{1}
746   \let\pstart\pstartL
747   \let\thepstart\thepstartL
748   \let\pend\pendL
749   \let\memorydump\memorydumpL
750   \Leftsidehook
751   \let\old@startstanza\@startstanza
752   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
753 }{
754   \Leftsidehookend}
755 %

```

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

`\Leftsidehookend`

`\Rightsidehook`

`\Rightsidehookend`

```

756 \newcommand*{\Leftsidehook}{}
757 \newcommand*{\Leftsidehookend}{}
758 \newcommand*{\Rightsidehook}{}
759 \newcommand*{\Rightsidehookend}{}
760
761 %

```

**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.

```

762 \newenvironment{Rightside}{%
763   \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
764     \led@err@Rightside@PreviousNotPrinted%
765   \fi%
766   \ledRcoltrue
767   \let\beginnumbering\beginnumberingR
768   \let\endnumbering\endnumberingR
769   \let\pausenumbering\pausenumberingR
770   \let\resumenumbering\resumenumberingR
771   \let\memorydump\memorydumpR
772   \let\thepstart\thepstartR
773   \let\pstart\pstartR
774   \let\pend\pendR
775   \let\ledpb\ledpbR
776   \let\lednopb\lednopbR
777   \let\lineation\lineationR
778   \Rightsidehook
779   \let\old@startstanza\@startstanza
780   \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
781 }{%
782   \ledRcolfalse
783   \Rightsidehookend
784 }
785
786 %

```

## 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.

```
787 \newcount\num@linesR
788 \newbox\one@lineR
789 \newcount\par@lineR
790 %
```

`\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. `\pstartR` `\pstart` needs to appear at the start of every paragraph that is to be numbered.

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

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

```
791
792 \newcounter{pstartL}
793 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
794 \newcounter{pstartR}
795 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
796
797 \newcommandx*{\pstartL}[1][1]{%
798   \if@nobreak%
799     \let\@oldnobreak\@nobreaktrue%
800   \else%
801     \let\@oldnobreak\@nobreakfalse%
802   \fi%
803   \@nobreaktrue%
804   \ifluatex%
805     \xdef\l@luatextextdir@L{\the\textdir}%
806     \xdef\l@luatexpardir@L{\the\pardir}%
807     \xdef\l@luatexbodydir@L{\the\bodydir}%
808   \fi%
809   \ifnumbering \else%
810     \led@err@PstartNotNumbered%
811     \beginnumbering%
812   \fi%
813   \ifnumberedpar@%
814     \led@err@PstartInPstart%
```



```

815 \pend%
816 \fi%
817 %

```

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

```

818 \ifpst@rtedL\else%
819 \list@clear{\inserts@list}%
820 \global\let\next@insert=\empty%
821 \global\pst@rtedLtrue%
822 \fi%
823 \begingroup\normal@pars%
824 %

```

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.

```

825 \global\advance\l@dnumpsstartsL \@ne%
826 \ifnum\l@dnumpsstartsL>\l@dc@maxchunks%
827 \led@err@TooManyPstarts%
828 \global\l@dnumpsstartsL=\l@dc@maxchunks%
829 \fi%
830 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpsstartsL}=\vbox\bgroup%
831 %

```

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.

```

832 \l@dzeropenalties%
833 \ifautopar\else%
834 \ifnumberpstart%
835 \ifsidepstartnum%
836 \else%
837 \thepstartL%
838 \fi%
839 \fi%
840 \fi%
841 \hsize=\Lcolwidth%
842 \numberedpar@true%
843 \iflabelpstart\protected@edef\@currentlabel%
844 {\p@pstartL\thepstartL}\fi%
845 %

```

Dump the optional arguments

```

846 \ifstrempy{#1}%
847 {\csgdef{before@pstartL@the\l@dnumpsstartsL}{\at@every@pstart}}%
848 {\csgdef{before@pstartL@the\l@dnumpsstartsL}{\noindent#1}}%
849 \at@every@pstart@call%
850 %

```

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

```
851 \ignorespaces%
```

```
852 %
```

```
853 }
```

```
854 %
```

The same for right side.

```
855 \newcommandx*{\pstartR}[1][1]{%
```

```
856 \if@nbreak%
```

```
857 \let\@oldnbreak\@nbreaktrue%
```

```
858 \else%
```

```
859 \let\@oldnbreak\@nbreakfalse%
```

```
860 \fi%
```

```
861 \@nbreaktrue%
```

```
862 \ifluatex%
```

```
863 \xdef\l@luatextextdirR{\the\textdir}%
```

```
864 \xdef\l@luatexpardirR{\the\pardir}%
```

```
865 \xdef\l@luatexbodydirR{\the\bodydir}%
```

```
866 \fi%
```

```
867 \ifnumberingR \else%
```

```
868 \led@err@PstartNotNumbered%
```

```
869 \beginnumberingR%
```

```
870 \fi%
```

```
871 \ifnumberedpar@%
```

```
872 \led@err@PstartInPstart%
```

```
873 \pendR%
```

```
874 \fi%
```

```
875 \ifpst@rtedR\else%
```

```
876 \list@clear{\inserts@listR}%
```

```
877 \global\let\next@insertR=\empty%
```

```
878 \global\pst@rtedRtrue%
```

```
879 \fi%
```

```
880 \begingroup\normal@pars%
```

```
881 \global\advance\l@dnumpsstartsR \@ne%
```

```
882 \ifnum\l@dnumpsstartsR>\l@dc@maxchunks%
```

```
883 \led@err@TooManyPstarts%
```

```
884 \global\l@dnumpsstartsR=\l@dc@maxchunks%
```

```
885 \fi%
```

```
886 \global\setnamebox{l@dRcolrawbox\the\l@dnumpsstartsR}=\vbox\bgroup%
```

```
887 \l@dzeropenalties%
```

```
888 \ifautopar\else%
```

```
889 \ifnumberpstart%
```

```
890 \ifsidepstartnum\else%
```

```
891 \thepstartR%
```

```
892 \fi%
```

```
893 \fi%
```

```
894 \fi%
```

```
895 \hsize=\Rcolwidth%
```

```
896 \numberedpar@true%
```

```
897 \iflabelpstart\protected@edef\@currentlabel%
```

```

898     {\p@pstartR\thepstartR}\fi%
899     \ifstrempy{#1}%
900     {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
901     {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
902     \at@every@pstart@call%
903     \ignorespaces%
904   }
905 %

```

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

```

906 \newcommand*{\pendL}[1][1]{%
907   \ifnumbering \else%
908     \led@err@PendNotNumbered%
909   \fi%
910   \ifnumberedpar@ \else%
911     \led@err@PendNoPstart%
912   \fi%
913 %

```

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.

```

914   \endgraf\global\num@lines=\prevgraf\egroup%
915   \global\par@line=0%
916 %

```

End the group that was begun in the \pstart.

```

917   \endgroup%
918   \ignorespaces%
919   \@oldnobreak%
920   \dump@pstartL@pc%
921   \ifnumberpstart%
922     \addtocounter{pstartL}{1}%
923   \fi
924   \parledgroup@beforenotes@save{L}%
925 %

```

Dump content of the optional argument.

```

926   \ifstrempy{#1}%
927   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
928   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
929   }
930 %

```

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

```

931 \newcommand*{\pendR}[1][1]{%
932   \ifnumberingR \else%

```

```

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

```

**\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.

```

954 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call{#1}}%
955 \gdef\at@every@pstart@call{}%
956 %

```

**\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**

```

957 \newif\ifprint@last@after@pendL%
958 \newif\ifprint@last@after@pendR%
959 %

```

## 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**

```

960 \newbox\l@dleftbox
961 \newbox\l@drightbox
962
963 %

```

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

```
\countRline
964 \newcount\countLline
965 \countLline \z@
966 \newcount\countRline
967 \countRline \z@
968
969 %
```

`\@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
970 \newcount\@donereallinesL
971 \newcount\@donetotallinesL
972 \newcount\@donereallinesR
973 \newcount\@donetotallinesR
974
975 %
```

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

```
976 \newcommand*{\do@lineL}{%
977 \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscL}%
978 \advance\countLline \@ne%
979 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscL}%
980 {\vbadness=10000%
981 \splittopskip=\z@%
982 \do@lineLhook%
983 \l@demtyd@ta%
984 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscL}%
985 to\baselineskip}%
986 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
\parledgroup@notes@startL}{}%
987 \unvbox\one@line \global\setbox\one@line=\lastbox%
988 \@writepageofparL%
989 \getline@numL%
990 \ifnum\@lock>\@ne%
991 \inserthangingsymboltrue%
992 \else%
993 \inserthangingsymbolfalse%
994 \fi%
995 \setbox\l@dleftbox%
996 \hb@xt@ \Lcolwidth{%
997 \ifl@dhiddenumber%
998 \global\l@dhiddenumberfalse%
999 \f@x@l@cks%
1000 \else%
1001 \affixline@num%
```

```

1002 \fi%
1003 \xifinlist{\the\l@dpscl}{\eled@sections@@}%
1004 {\add@inserts\affixside@note}%
1005 {\print@lineL}%
1006 }%
1007 \add@penaltiesL%
1008 \global\advance\@donereallinesL\@ne%
1009 \global\advance\@donetotallinesL\@ne%
1010 \else%
1011 \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
1012 \global\advance\@donetotallinesL\@ne%
1013 \fi%
1014 }%
1015
1016
1017 %

```

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

```

1018 \def\print@lineL{%
1019 \affixpstart@numL%
1020 \l@dld@ta %space kept for backward compatibility
1021 \add@inserts\affixside@note%
1022 \l@dlsn@te %space kept for backward compatibility
1023 \hb@xt@ \Lcolwidth{\ledllfill\hb@xt@ \wd\one@line{%
1024 \do@insidelineLhook%
1025 \ifluatex%
1026 \textdir\l@luatextextdir@L%
1027 \fi%
1028 \new@lineL%
1029 \inserthangingsymbolL%
1030 \l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%
1031 \l@drsn@te}}
1032
1033 %

```

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

```

1034 \def\print@eledsectionL{%%
1035 \addtocounter{pstartL}{-1}%
1036 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{ }
1037 \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}{ }
1038 \numdef{\temp@}{\l@dpscl-1}%
1039 \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1040 \@eled@sectioningtrue%
1041 \bgroup%
1042 \ifluatex%
1043 \textdir\l@luatextextdir@L%
1044 \pardir\l@luatexpardir@L%

```

```

1045 \bodydir\l@uatexbodydir@L%
1046 \ifdefstring{\l@uatextextdir@L}{TRT}{\@RTLtrue}{}%
1047 \fi%
1048 \csuse{eled@sectioning@the\l@dpscL}%
1049 \egroup%
1050 \@eled@sectioningfalse%
1051 \global\csundef{eled@sectioning@the\l@dpscL}%
1052 \if@RTL%
1053 \hspace{-3\paperwidth}%
1054 {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1055 \else%
1056 \hspace{3\paperwidth}%
1057 {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1058 \fi%
1059 \vskip\eledsection@correcting@skip%
1060 }
1061
1062 %

```

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

```

\doinssidelineLhook 1063 \newcommand*\dolineLhook[1]{\gdef\do@lineLhook{#1}}%
\doinssidelineRhook 1064 \newcommand*\dolineRhook[1]{\gdef\do@lineRhook{#1}}%
1065 \newcommand*\doinssidelineLhook[1]{\gdef\do@insidelineLhook{#1}}%
1066 \newcommand*\doinssidelineRhook[1]{\gdef\do@insidelineRhook{#1}}%
1067
1068 %

```

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

```

\dolineLhook 1069 \newcommand*\do@lineLhook{}
\dolineRhook 1070 \newcommand*\do@lineRhook{}
\doinssidelineLhook 1071 \newcommand*\do@insidelineLhook{}
\doinssidelineRhook 1072 \newcommand*\do@insidelineRhook{}
1073
1074 %

```

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

```

1075 \newcommand*\do@lineR{%
1076 \let\linenumrep\linenumrepR%
1077 \let\sublinenumrep\sublinenumrepR%
1078 \letcs{ifnumberpstart}{numberpstart@R\the\l@dpscR}%
1079 \ledRcol@true%
1080 \advance\countRline \@ne%
1081 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscR}%
1082 {\vbadness=10000%
1083 \splittopskip=\z@%

```

```

1084 \do@lineRhook%
1085 \l@emptyd@ta%
1086 \global\setbox\one@lineR=\vsplit\namebox{l@dRcolrawbox\the\l@dpscR}%
1087 to\baselineskip}%
1088 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
parledgroup@notes@startR}{}%
1089 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
1090 \@writepageofparR%
1091 \getline@numR%
1092 \ifnum\@lockR>\@ne%
1093 \inserthangingsymbolRtrue%
1094 \else%
1095 \inserthangingsymbolRfalse%
1096 \fi%
1097 \setbox\l@drightbox%
1098 \hb@xt@ \Rcolwidth{%
1099 \ifl@dhiddenumber%
1100 \global\l@dhiddenumberfalse%
1101 \f@x@l@cksR%
1102 \else%
1103 \affixline@numR%
1104 \fi%
1105 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1106 {\add@insertsR\affixside@noteR}%
1107 {\print@lineR}%
1108 }%
1109 \add@penaltiesR%
1110 \global\advance\@donereallinesR\@ne%
1111 \global\advance\@donetotallinesR\@ne%
1112 \else%
1113 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*\Rcolwidth}%
1114 \global\advance\@donetotallinesR\@ne%
1115 \fi%
1116 \ledRcol@false%
1117 }
1118
1119
1120 %

```

```

\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.

```

1121 \newcommand*\getline@numR{%
1122 \global\advance\absline@numR \@ne
1123 \do@actionsR
1124 \do@ballastR

```



```

1125 \ifledgroupnotesR\else
1126   \ifnumberline
1127     \ifsublines@
1128       \ifnum\sub@lockR<\tw@
1129         \global\advance\subline@numR \@ne
1130       \fi
1131     \else
1132       \ifnum\@lockR<\tw@
1133         \global\advance\line@numR \@ne
1134         \global\subline@numR \z@
1135       \fi
1136     \fi
1137   \fi
1138 \fi
1139 }
1140 \newcommand*{\getline@numL}{%
1141   \global\advance\absline@num \@ne
1142   \do@actions
1143   \do@ballast
1144   \ifledgroupnotesL\else
1145     \ifnumberline
1146       \ifsublines@
1147         \ifnum\sub@lock<\tw@
1148           \global\advance\subline@num \@ne
1149         \fi
1150       \else
1151         \ifnum\@lock<\tw@
1152           \global\advance\line@num \@ne
1153           \global\subline@num \z@
1154         \fi
1155       \fi
1156     \fi
1157   \fi
1158 }
1159
1160
1161 %

```

**\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.

```

1162 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1163   \begingroup
1164     \advance\absline@numR \@ne
1165     \ifnum\next@actionlineR=\absline@numR
1166       \ifnum\next@actionR>-1001
1167         \global\advance\ballast@count by -\c@ballast
1168       \fi
1169     \fi
1170   \endgroup}

```

1171 %

`\l@dskipversenumberR` The `\do@actionsR` macro looks at the list of actions to take at particular right text  
`\do@actionsR` 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  
`\do@actions@nextR` this.

```

1172
1173 \newif\ifl@dskipversenumberR
1174 \newcommand*{\do@actions@fixedcodeR}{%
1175   \ifcase\@l@dttempcnta%
1176   \or% % 1001
1177     \global\sublines@true
1178   \or% % 1002
1179     \global\sublines@false
1180   \or% % 1003
1181     \global\@lockR=\@ne
1182   \or% % 1004%
1183     \ifnum\@lockR=\tw@
1184       \global\@lockR=\thr@@
1185     \else
1186       \global\@lockR=\z@
1187     \fi
1188   \or% % 1005
1189     \global\sub@lockR=\@ne
1190   \or% % 1006
1191     \ifnum\sub@lockR=\tw@
1192       \global\sub@lockR=\thr@@
1193     \else
1194       \global\sub@lockR=\z@
1195     \fi
1196   \or% % 1007
1197     \l@dskipnumbertrue
1198   \or% % 1008
1199     \l@dskipversenumberRtrue%
1200   \or% % 1009
1201     \l@dhidenumbertrue%
1202   \else%
1203     \led@warn@BadAction
1204   \fi%
1205 }
1206
1207
1208 \newcommand*{\do@actionsR}{%
1209   \global\let\do@actions@nextR=\relax
1210   \@l@dttempcntb=\absline@numR
1211   \ifnum\@l@dttempcntb<\next@actionlineR\else
1212     \ifnum\next@actionR>-1001\relax
1213       \global\page@numR=\next@actionR
1214       \ifbypage@R

```

```

1215 \global\line@numR \z@ \global\subline@numR \z@
1216 \fi
1217 \else
1218 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1219 \@l@dttempcnta=-\next@actionR
1220 \advance\@l@dttempcnta by -5001\relax
1221 \ifsublines@
1222 \global\subline@numR=\@l@dttempcnta
1223 \else
1224 \global\line@numR=\@l@dttempcnta
1225 \fi
1226 \else
1227 \@l@dttempcnta=-\next@actionR
1228 \advance\@l@dttempcnta by -1000\relax
1229 \do@actions@fixedcodeR
1230 \fi
1231 \fi
1232 \ifx\actionlines@listR\empty
1233 \gdef\next@actionlineR{1000000}%
1234 \else
1235 \gl@p\actionlines@listR\to\next@actionlineR
1236 \gl@p\actions@listR\to\next@actionR
1237 \global\let\do@actions@nextR=\do@actionsR
1238 \fi
1239 \fi
1240 \do@actions@nextR}
1241
1242 %

```

## VIII.4 Line number printing

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

```

1243 \ch@cksub@l@ckR
1244 \ch@ck@l@ckR
1245 \f@x@l@ckR
1246 \affixline@numR
1247
1248 \newcommand*{\l@dcalcnm}[3]{%
1249 \ifnum #1 > #2\relax
1250 \@l@dttempcnta = #1\relax
1251 \advance\@l@dttempcnta by -#2\relax
1252 \divide\@l@dttempcnta by #3\relax
1253 \multiply\@l@dttempcnta by #3\relax
1254 \advance\@l@dttempcnta by #2\relax
1255 \else
1256 \@l@dttempcnta=#2\relax
1257 \fi}
1258
1259 \newcommand*{\ch@cksub@l@ckR}{%
1260 \ifcase\sub@lockR
1261 \or
1262 \ifnum\sublock@disp=\@ne

```

```

1259 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1260 \fi
1261 \or
1262 \ifnum\sublock@disp=\tw@
1263 \else
1264 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1265 \fi
1266 \or
1267 \ifnum\sublock@disp=\z@
1268 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1269 \fi
1270 \fi}
1271
1272 \newcommand*{\ch@ck@l@ckR}{%
1273 \ifcase\@lockR
1274 \or
1275 \ifnum\lock@disp=\@ne
1276 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1277 \fi
1278 \or
1279 \ifnum\lock@disp=\tw@
1280 \else
1281 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1282 \fi
1283 \or
1284 \ifnum\lock@disp=\z@
1285 \l@l@dttempcntb \z@ \l@l@dttempcnta \@ne
1286 \fi
1287 \fi}
1288
1289 \newcommand*{\f@x@l@cksR}{%
1290 \ifcase\@lockR
1291 \or
1292 \global\@lockR \tw@
1293 \or \or
1294 \global\@lockR \z@
1295 \fi
1296 \ifcase\sub@lockR
1297 \or
1298 \global\sub@lockR \tw@
1299 \or \or
1300 \global\sub@lockR \z@
1301 \fi}
1302
1303
1304 \newcommand*{\affixline@numR}{%
1305 \ifledgroupnotesR\else\ifnumberline
1306 \ifl@dskipnumber
1307 \global\l@dskipnumberfalse
1308 \else

```

```

1309 \ifsublines@
1310 \l@dttempcntb=\subline@numR
1311 \l@dcalcnnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR
}%
1312 \ch@cksub@lockR
1313 \else
1314 \l@dttempcntb=\line@numR
1315 \ifx\linenumberlist\empty
1316 \l@dcalcnnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1317 \else
1318 \l@dttempcnta=\line@numR
1319 \edef\rem@inder{,\linenumberlist,\number\line@numR,}%
1320 \edef\sc@n@list{\def\noexpand\sc@n@list
1321 ###1,\number\l@dttempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1322 \sc@n@list\expandafter\sc@n@list\rem@inder|}%
1323 \ifx\rem@inder\empty\advance\l@dttempcnta\@ne\fi
1324 \fi
1325 \ch@ck@l@ckR
1326 \fi
1327 \ifnum\l@dttempcnta=\l@dttempcntb
1328 \ifl@dskipversenumberR\else
1329 \if@twocolumn
1330 \if@firstcolumn
1331 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1332 \else
1333 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1334 \fi
1335 \else
1336 \l@dttempcntb=\line@marginR
1337 \ifnum\l@dttempcntb>\@ne
1338 \advance\l@dttempcntb by\page@numR
1339 \fi
1340 \ifodd\l@dttempcntb
1341 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1342 \else
1343 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1344 \fi
1345 \fi
1346 \fi
1347 \fi
1348 \f@x@l@cksR
1349 \fi
1350 \fi
1351 \fi}
1352 %

```

## 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@numL1353
\affixpstart@numR1354 \newcommand*{\affixpstart@numL}{%
  \leftpstartnumR1355 \ifsidepstartnum
  \rightpstartnumR1356 \if@twocolumn
    \if@firstcolumn
      \leftpstartnumL1357
      \rightpstartnumL1358 \gdef\l@dld@ta{\llap{\leftpstartnumL}}%
      \ifpstartnumR1359
        \gdef\l@drd@ta{\rlap{\rightpstartnumL}}%
        \fi
      \else
        \l@dttempcntb=\line@margin
        \ifnum\l@dttempcntb>\@ne
          \advance\l@dttempcntb \page@num
        \fi
        \ifodd\l@dttempcntb
          \gdef\l@drd@ta{\rlap{\rightpstartnumL}}%
        \else
          \gdef\l@dld@ta{\llap{\leftpstartnumL}}%
        \fi
      \fi
    \fi
  }
  \newcommand*{\affixpstart@numR}{%
    \ifsidepstartnum
    \if@twocolumn
      \if@firstcolumn
        \gdef\l@dld@ta{\llap{\leftpstartnumR}}%
      \else
        \gdef\l@drd@ta{\rlap{\rightpstartnumR}}%
        \fi
      \else
        \l@dttempcntb=\line@marginR
        \ifnum\l@dttempcntb>\@ne
          \advance\l@dttempcntb \page@numR
        \fi
        \ifodd\l@dttempcntb
          \gdef\l@drd@ta{\rlap{\rightpstartnumR}}%
        \else
          \gdef\l@dld@ta{\llap{\leftpstartnumR}}%
        \fi
      \fi
    \fi
  }
}

```

```

1397 \newcommand*{\leftpstartnumL}{
1398 \ifpstartnum
1399 \thepstartL
1400 \kern\linenumsep\global\pstartnumfalse\fi
1401 }
1402 \newcommand*{\rightpstartnumL}{
1403 \ifpstartnum\kern\linenumsep
1404 \thepstartL
1405 \global\pstartnumfalse\fi
1406 }
1407 \newif\ifpstartnumR
1408 \pstartnumRtrue
1409 \newcommand*{\leftpstartnumR}{
1410 \ifpstartnumR
1411 \thepstartR
1412 \kern\linenumsep\global\pstartnumRfalse\fi
1413 }
1414 \newcommand*{\rightpstartnumR}{
1415 \ifpstartnumR\kern\linenumsep
1416 \thepstartR
1417 \global\pstartnumRfalse\fi
1418 }
1419 %

```

## 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.

```

1420 \list@create{\inserts@listR}
1421 %

```

`\add@insertsR` The right text version.

`\add@inserts@nextR`

```

1422 \newcommand*{\add@insertsR}{%
1423 \global\let\add@inserts@nextR=\relax
1424 \ifx\inserts@listR\empty \else
1425 \ifx\next@insertR\empty
1426 \ifx\insertlines@listR\empty
1427 \global\noteschanged@true
1428 \gdef\next@insertR{100000}%
1429 \else
1430 \gl@p\insertlines@listR\to\next@insertR
1431 \fi
1432 \fi
1433 \ifnum\next@insertR=\absline@numR
1434 \gl@p\inserts@listR\to\@insertR
1435 \@insertR
1436 \global\let\@insertR=\undefined

```

```

1437 \global\let\next@insertR=\empty
1438 \global\let\add@inserts@nextR=\add@insertsR
1439 \fi
1440 \fi
1441 \add@inserts@nextR}
1442
1443 %

```

### 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.

```
1444 \newcommand*{\add@penaltiesL}{%
1445 \newcommand*{\add@penaltiesR}{%
1446
1447 %
```

## 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.

```
1448 \newcommand*{\flush@notesR}{%
1449 \xloop
1450 \ifx\inserts@listR\empty \else
1451 \gl@p\inserts@listR\to\@insertR
1452 \@insertR
1453 \global\let\@insertR=\undefined
1454 \repeat}
1455
1456 %
```

## 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. 82 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
```

```
1457 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1458 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1459 \ifl@d@pnum #1\fullstop\fi
1460 \ifledplinenum \linenumr@p{#2}\@Rlineflag\else \sympplinenum\fi
1461 \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1462 \ifl@d@dash \endashchar\fi
1463 \ifl@d@pnum #4\fullstop\fi
1464 \ifl@d@elin \linenumr@p{#5}\@Rlineflag\fi
1465 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
```

```

1466 \endgroup}
1467
1468
1469 %

```

## IX.2 Footnotes output specific to \Pages

```

\print@Xnotes@forpages
\correct@Xfootins@box
\print@notesX@forpages
\correct@footinsX@box

```

The `\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 `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed<sup>6</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.
- 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:

---

<sup>6</sup>See <http://tex.stackexchange.com/a/230332/7712>.

- 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  $\text{\TeX}$  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  $\text{\TeX}$  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  $\text{\TeX}$  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`.

```
1470 \newcommand\print@Xnotes@forpages[1]{%
1471 %
```

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

```
1472 \ifcseempty{Xonlyside@#1}{%
1473 \csuse{#1footstart}{#1}%
1474 \csuse{#1footgroup}{#1}%
1475 }%
1476 %
```

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

```
1477 {%
1478 \ifboolexpr{%
1479 ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\ifnumodd{\c@page
1480 }})%
1481 or%
1482 (test {\ifcsstring{Xonlyside@#1}{R}} and test{\ifnumodd{\c@page}}))%
1483 }%
1484 %
```

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

```
1484 {%
1485 \correct@Xfootins@box{#1}%
1486 \csuse{#1footstart}{#1}%
1487 \csuse{#1footgroup}{#1}%
1488 %
```

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

```

1489 \global\count\csuse{#1footins}=0%
1490 \global\skip\csuse{#1footins}=0pt%
1491 %

```

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

```

1492 \csuse{Xnotefontsize@#1}%
1493 \global\advance\dimen\csuse{#1footins} by -\baselineskip%
1494 %

```

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

```

1495 \global\boolfalse{keepforXside@#1}%
1496 }%
1497 %

```

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.

```

1498 {%
1499 \global\booltrue{keepforXside@#1}%
1500 %

```

Then restore expected rooms for notes on the next page.

```

1501 \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1502 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
1503 %

```

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

```

1504 \bgroup%
1505 \csuse{Xnotefontsize@#1}%
1506 \global\advance\dimen\csuse{#1footins} by \baselineskip%
1507 \egroup%
1508 %

```

```

1509 % End of \protect\cs{print@Xnotes@forpages}.
1510 }%
1511 }%
1512 }%
1513 %

```

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

```

1514 \newcommand{\correct@Xfootins@box}[1]{%
1515 %

```

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.

```

1516 \ifbool{keepforXside@#1}{%
1517 %

```

Some setting needed to do the right splitting.

```
1518 \csuse{Xnotefontsize@#1}%
1519 \splittopskip=0pt%
1520 %
```

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

```
1521 \global\setbox\csuse{#1footins}=\vbox{%
1522 \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1523 \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1524 \unvbox\csuse{#1footins}%
1525 }%
1526 %
```

End of the macro.

```
1527 }{}%
1528 }%
1529 %
```

And now, the same for familiar footnotes.

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

```

1559     }%
1560   }%
1561 }%
1562 \newcommand{\correct@footinsX@box}[1]{%
1563   \ifbool{keepforsideX@#1}{%
1564     \csuse{notefontsizeX@#1}%
1565     \splittopskip=0pt%
1566     \global\setbox\csuse{footins#1}=\vbox{%
1567       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1568       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1569       \unvbox\csuse{footins#1}%
1570     }%
1571   }{}%
1572 }%
1573 %

```

## 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.

```

1574 \list@create{\labelref@listR}
1575
1576 %

```

**\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.

```

1577 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1578   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1579     \led@warn@DuplicateLabel{#4}%
1580   \fi
1581   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\@Rlineflag|#3|#4}%
1582   \ignorespaces}
1583 \AtBeginDocument{%
1584   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1585   }
1586
1587 %

```

**\@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*
1588 \WithSuffix\newcommand\sidenotemargin*[1]{%
1589   \l@dgetsidenote@margin{#1}
1590   \global\sidenote@marginR=\@l@tempcntb
1591   \global\sidenote@margin=\@l@tempcntb
1592 }
1593 \newcount\sidenote@marginR
1594 \global\sidenote@margin=\@ne
1595
1596 %
```

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

```
1597 \newcommand*{\affixside@noteR}{%
1598   \def\sidenotecontent@{}%
1599   \numgdef{\itemcount@}{0}%
1600   \def\do##1{%
1601     \ifnumequal{\itemcount@}{0}%
1602       {%
1603         \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
1604       {\appto\sidenotecontent@{\sidenotessep ##1}%
1605       }%
1606       \numgdef{\itemcount@}{\itemcount@+1}%
1607     }%
1608     \dolistloop{\l@dcnotesetext}%
1609     \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1610   \gdef\@templ@d{}%
1611   \gdef\@templ@n{\l@dcnotesetext\l@dcnotesetext@1\l@dcnotesetext@r}%
1612   \ifx\@templ@d\@templ@n \else%
1613     \if@twocolumn%
1614       \if@firstcolumn%
1615         \setl@dlp@rbox{##1}{\sidenotecontent@}%
1616       \else%
1617         \setl@drp@rbox{\sidenotecontent@}%
1618       \fi%
1619     \else%
1620       \@l@tempcntb=\sidenote@marginR%
1621       \ifnum\@l@tempcntb>\@ne%
1622         \advance\@l@tempcntb by\page@numR%
1623       \fi%
1624       \ifodd\@l@tempcntb%
1625         \setl@drp@rbox{\sidenotecontent@}%
1626       \gdef\sidenotecontent@{}%

```

```

1627 \numdef{\itemcount@}{0}%
1628 \dolistloop{\l@dcsnotetext@l}%
1629 \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1630 \setl@dlp@rbox{\sidenotecontent@}%
1631 \else%
1632 \setl@dlp@rbox{\sidenotecontent@}%
1633 \gdef\sidenotecontent@{}%
1634 \numdef{\itemcount@}{0}%
1635 \dolistloop{\l@dcsnotetext@r}%
1636 \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1637 \setl@drp@rbox{\sidenotecontent@}%
1638 \fi%
1639 \fi%
1640 \fi%
1641 }
1642
1643 %

```

## 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.

```

\inserthangingsymbolL44 \newif\ifinserthangingsymbolR
\inserthangingsymbolR45 \newcommand{\inserthangingsymbolL}{%
1646 \ifinserthangingsymbol%
1647 \ifinstanzaL%
1648 \hskip \@ifundefined{sza@00}{0}{\expandafter%
1649 \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1650 \@hangingsymbol%
1651 \fi%
1652 \fi%
1653 }%
1654 \newcommand{\inserthangingsymbolR}{%
1655 \ifinserthangingsymbolR%
1656 \ifinstanzaR%
1657 \hskip \@ifundefined{sza@00}{0}{\expandafter%
1658 \noexpand\csname sza@00\endcsname}\stanzaindentbase%

```



```

1659 \hangingsymbol%
1660 \fi%
1661 \fi%
1662 }%
1663 %

```

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.

```

1664 \chardef\next=\catcode`&
1665 \catcode`&=\active
1666
1667 %

```

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

```

1668 \newenvironment{astanza}[1][1]{%
1669 \catcode`&=\active
1670 \global\stanza@count\@ne\stanza@modulo\@ne
1671 \ifnum\usernamecount{sza@0@}=\z@
1672 \let\stanza@hang\relax
1673 \let\endlock\relax
1674 \else
1675 \rightskip\z@ plus 1fil\relax
1676 \fi
1677 \ifnum\usernamecount{szp@0@}=\z@
1678 \let\sza@penalty\relax
1679 \fi
1680 \def&{%
1681 \endlock\mbox{}%
1682 \sza@penalty
1683 \global\advance\stanza@count\@ne
1684 \@astanza@line}%
1685 \def\&{\@stopastanza}%
1686 \pstart[#1]%
1687 \@astanza@line
1688 \let\par\relax\ignorespaces%No paragraph in verses
1689 }{}
1690
1691 %

```

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

```

1692 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1693 \endlock\mbox{}%
1694 \pend[#1]%
1695 }%
1696 %

```

`\@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.

```

1697 \newcommand*{\@astanza@line}{%
1698   \ifnum\value{stanzaindentsrepetition}=0
1699     \parindent=\csname sza@\number\stanza@count
1700               @\endcsname\stanzaindentbase
1701   \else
1702     \parindent=\csname sza@\number\stanza@modulo
1703               @\endcsname\stanzaindentbase
1704     \managestanza@modulo
1705   \fi
1706   \endgraf
1707   \stanza@hang%
1708   \ignorespaces}
1709
1710 %

```

Lastly reset the modified category codes.

```

1711 \catcode`\&=\next
1712
1713 %

```

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

```

\thestanzaR
1714 \newcounter{stanzaL}
1715 \newcounter{stanzaR}
1716 \renewcommand{\thestanzaL}{%
1717   \textbf{\arabic{stanzaL}}%
1718 }
1719 \renewcommand{\thestanzaR}{%
1720   \textbf{\arabic{stanzaR}}%
1721 }
1722 %
1723 %

```

## XIV Naming macros

The  $\text{\LaTeX}$  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
1724 \providecommand*{\newnamebox}[1]{%
1725   \expandafter\newbox\csname #1\endcsname}
1726 \providecommand*{\setnamebox}[1]{%

```

```

1727 \expandafter\setbox\csname #1\endcsname}
1728 \providecommand*{\unhnamebox}[1]{%
1729 \expandafter\unhbox\csname #1\endcsname}
1730 \providecommand*{\unvnamebox}[1]{%
1731 \expandafter\unvbox\csname #1\endcsname}
1732 \providecommand*{\namebox}[1]{%
1733 \csname #1\endcsname}
1734
1735 %

```

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

```

\usernamecount
1736 \providecommand*{\newnamecount}[1]{%
1737 \expandafter\newcount\csname #1\endcsname}
1738 \providecommand*{\usernamecount}[1]{%
1739 \csname #1\endcsname}
1740
1741 %

```

## 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
1742 \newif\ifl@dusedbabel
\l@dusedbabeltrue
1743 %

```

`\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 `\`.

```

1744 \patchcmd{\bbl@set@language}%
1745 {\select@language{\language}}%
1746 {\edef\language{#1}\select@language{\language}}%

```

```

1747 {}%
1748 {}%
1749
1750 %

```

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
1751 \newcommand*{\l@duselanguage}[1]{}
1752 \gdef\theledlanguageL{}
1753 \gdef\theledlanguageR{}
1754
1755 %

```

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

```

1756 \AtBeginDocument{%
1757   \ifundefined{xpg@main@language}{%
1758     \ifundefined{bbl@main@language}{%
1759

```

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

```

1760   \l@dusedbabelfalse
1761 }{%
1762 %

```

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.

```

1763   \l@dusedbabeltrue
1764   \let\l@doldselectlanguage\selectlanguage
1765   \let\l@doldbbl@set@language\bbl@set@language
1766   \renewcommand{\selectlanguage}[1]{%
1767     \l@doldselectlanguage{#1}%
1768     \ifledRcol \gdef\theledlanguageR{#1}%
1769     \else      \gdef\theledlanguageL{#1}%
1770     \fi}
1771 %

```

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

```

1772   \renewcommand*{\l@duselanguage}[1]{%
1773     \l@doldselectlanguage{#1}}
1774 %

```

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

```

1775 \gdef\theledlanguageL{\bbl@main@language}%
1776 \gdef\theledlanguageR{\bbl@main@language}%
1777 }%
1778 }
1779 %

```

If use polyglossia

```

1780 { \let\old@otherlanguage\otherlanguage%
1781 \renewcommand{\otherlanguage}[2] [] {%
1782 \selectlanguage[#1]{#2}%
1783 \ifledRcol \gdef\theledlanguageR{#2}%
1784 \else \gdef\theledlanguageL{#2}%
1785 \fi}%
1786 \let\l@ducelanguage\select@language%
1787 \gdef\theledlanguageL{\xpg@main@language}%
1788 \gdef\theledlanguageR{\xpg@main@language}%
1789 %

```

That is it.

```

1790 }}
1791 %

```

## 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 5120 chunk pairs.

```

1792 \newcount\l@dc@maxchunks
1793 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1794 \maxchunks{5120}
1795
1796 %

```

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

```

\l@dnumpstartsR
1797 \newcount\l@dnumpstartsR
1798
1799 %

```

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

```

1800 \newcount\l@dpscL
1801 \newcount\l@dpscR
1802
1803 %

```

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

```

1804 \newcommand*\l@dsetuprawboxes{%
1805   \@l@tempcntb=\l@dc@maxchunks
1806   \loop\ifnum\@l@tempcntb>\z@
1807     \newnamebox{\l@dLcolrawbox\the\@l@tempcntb}
1808     \newnamebox{\l@dRcolrawbox\the\@l@tempcntb}
1809     \advance\@l@tempcntb \m@ne
1810   \repeat}
1811
1812 %

```

**\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.

```

1813 \newcommand*\l@dsetupmaxlinecounts{%
1814   \@l@tempcntb=\l@dc@maxchunks
1815   \loop\ifnum\@l@tempcntb>\z@
1816     \newnamecount{\l@dmaxlinesinpar\the\@l@tempcntb}
1817     \advance\@l@tempcntb \m@ne
1818   \repeat}
1819 \newcommand*\l@dzeromaxlinecounts{%
1820   \begingroup
1821   \@l@tempcntb=\l@dc@maxchunks
1822   \loop\ifnum\@l@tempcntb>\z@
1823     \global\usenamecount{\l@dmaxlinesinpar\the\@l@tempcntb}=\z@
1824     \advance\@l@tempcntb \m@ne
1825   \repeat
1826   \endgroup}
1827
1828 %

```

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

```

1829 \AtBeginDocument{%
1830   \l@dsetuprawboxes
1831   \l@dsetupmaxlinecounts
1832   \l@dzeromaxlinecounts
1833   \l@dnumpstartsL=\z@
1834   \l@dnumpstartsR=\z@
1835   \l@dpscL=\z@

```

```

1836 \l@dpscR=\z@}
1837
1838 %

```

## XVII Checking text to be processed

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

```

\@pstartstrue
\@pstartsfalse
\check@pstarts
1839 \newif\if@pstarts
1840 \newcommand*\check@pstarts}{%
1841 \@pstartsfalse
1842 \ifnum\l@dnumpstartsL>\l@dpscL
1843 \@pstartstrue
1844 \else
1845 \ifnum\l@dnumpstartsR>\l@dpscR
1846 \@pstartstrue
1847 \fi
1848 \fi
1849 }
1850
1851 %

```

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

```

\checkraw@text
1852 \newif\ifaraw@text
1853 \newcommand*\checkraw@text}{%
1854 \araw@textfalse
1855 \ifvbox\namebox{l@dLcolrawbox\the\l@dpscL}
1856 \araw@texttrue
1857 \else
1858 \ifvbox\namebox{l@dRcolrawbox\the\l@dpscR}
1859 \araw@texttrue
1860 \fi
1861 \fi
1862 }
1863
1864 %

```

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

```

1865 \newcommand*\@writelinesinparL}{%
1866 \edef\next{%
1867 \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1868 \next
1869 \global\@donereallinesL \z@}
1870 \newcommand*\@writelinesinparR}{%

```

```

1871 \edef\next{%
1872   \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1873 \next
1874 \global\@donereallinesR \z@}
1875
1876 %

```

`\@writepageofparL` These write the pages where start the first line of a chunk.

```

\@writepageofparR
1877 \newcommand*{\@writepageofparL}[0]{%
1878   \ifnum\@donereallinesL=\z@%
1879     \edef\next{%
1880       \write\linenum@out{\string\@pstart{\the\l@dpscl}{\the\c@page}{\the\
1881         numpagelinesL}}%
1882       }%
1883     \next%
1884   \fi%
1885 }%
1886 \newcommand*{\@writepageofparR}[0]{%
1887   \ifnum\@donereallinesR=\z@%
1888     \edef\next{%
1889       \write\linenum@outR{\string\@pstartR{\the\l@dpscR}{\the\c@page}{\the\
1890         numpagelinesR}}%
1891       }%
1892     \next%
1893   \fi%
1894 }%
1895 %

```

## XVIII Parallel columns

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

```

\@eledsectionR
1894 \newsavebox{\@eledsectionL}%
1895 \newsavebox{\@eledsectionR}%
1896 %

```

`\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.

```

1897 \newcommand*{\Columns}{%
1898   \ifl@dpairing%
1899     \led@err@Columns@InsideEnv%
1900   \fi%
1901   \l@dprintingcolumnstrue%
1902   \eledsection@correcting@skip=-\baselineskip% Correction for sections'
1903   titles
1904   \ifnum\l@dnumstartsL=\l@dnumstartsR\else
1905     \led@err@BadLeftRightPstarts{\the\l@dnumstartsL}{\the\l@dnumstartsR}%

```



```

1905 \fi
1906 %

```

Start a group and zero counters, etc.

```

1907 \begingroup
1908 \l@dzeropenalties
1909 \endgraf\global\num@lines=\prevgraf
1910 \global\num@linesR=\prevgraf
1911 \global\par@line=\z@
1912 \global\par@lineR=\z@
1913 \global\l@dpscL=\z@
1914 \global\l@dpscR=\z@
1915 %

```

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

```

1916 \check@pstarts
1917 \loop\if@pstarts
1918 \global\pstartnumtrue
1919 \global\pstartnumRtrue
1920 %

```

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.

```

1921 \global\advance\l@dpscL \@ne
1922 \global\advance\l@dpscR \@ne
1923 \restore@pstartL@pc%
1924 \restore@pstartR@pc%
1925 %

```

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

```

1926 \Columns@print@before@pstart%
1927 %

```

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

```

1928 \checkraw@text
1929 { \loop\ifaraw@text
1930 %

```

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

```

1931 \l@duselanguage{\theledlanguageL}%
1932 \do@lineL
1933 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1934 {%
1935 \ifdefstring{\@eledsectmark}{L}%
1936 {\csuse{eled@sectmark@the\l@dpscL}%
1937 }{}}%

```

```

1938      \global\csundef{eled@sectmark@the\l@dpscL}%
1939      \savebox{\@eledsectionL}{\parbox[t]{}[t]{\Lcolwidth}{\vbox
{} \print@eledsectionL}}}%\vbox{}-> prevent alignment troubles with RTL
language
1940      }%
1941      {}%
1942      \l@duselanguage{\theledlanguageR}%
1943      \do@lineR
1944      \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1945      {%
1946      \ifdefstring{\@eledsectmark}{R}%
1947      {\csuse{eled@sectmark@the\l@dpscR R}%
1948      }{}%
1949      \global\csundef{eled@sectmark@the\l@dpscR R}%
1950      \savebox{\@eledsectionR}{\parbox[t]{}[t]{\Rcolwidth}{\vbox
{} \print@eledsectionR}}}%\vbox{}-> prevent alignment troubles with RTL
language
1951      }%
1952      \hb@xt@ \hsize{%
1953      \ifdefstring{\columns@position}{L}{\}{\hfill }%
1954      \unhbox\l@dleftbox%
1955      \ifhbox\@eledsectionL%
1956      \usebox{\@eledsectionL}%
1957      \fi%
1958      \print@columnseparator%
1959      \unhbox\l@drightbox%
1960      \ifhbox\@eledsectionR%
1961      \usebox{\@eledsectionR}%
1962      \fi%
1963      \ifdefstring{\columns@position}{R}{\}{\hfill}%
1964      }%
1965      \checkraw@text
1966      \checkverseL
1967      \checkverseR
1968      \checkpb@columns
1969      \repeat}
1970 %

```

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.

```

1971      \@writelinesinparL
1972      \@writelinesinparR
1973      \check@pstarts
1974      \ifbypstart@%
1975      \write\linenum@out{\string\@set[1]}
1976      \resetprevline@
1977      \fi
1978      \ifbypstart@R
1979      \write\linenum@outR{\string\@set[1]}

```

```

1980         \resetprevline@
1981         \fi
1982         \Columns@print@after@pend%
1983     \repeat
1984 %

```

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.

```

1985     \flush@notes
1986     \flush@notesR
1987 \endgroup
1988 %

```

```

1989 \global\l@dpscl=\z@
1990 \global\l@dpscR=\z@
1991 \global\l@dnumpststartsL=\z@
1992 \global\l@dnumpststartsR=\z@
1993 \l@dprintingcolumnsfalse%
1994 \ignorespaces
1995 \global\instanzaLfalse
1996 \global\instanzaRfalse}
1997
1998 %

```

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

```

1999 \def\print@columnseparator{%
2000     \ifdim\beforecolumnseparator<0pt%
2001         \hfill%
2002     \else%
2003         \hspace{\beforecolumnseparator}%
2004     \fi%
2005     \columnseparator%
2006     \ifdim\aftercolumnseparator<0pt%
2007         \hfill%
2008     \else%
2009         \hspace{\beforecolumnseparator}%
2010     \fi%
2011 }%
2012 %

```

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

```

2013
2014 \newcommand{\checkpb@columns}{%
2015     \newif\if@pb

```

```

2016 \newif\if@nopb
2017 \IfStrEq{\led@pb@setting}{before}{
2018 \numdef{\next@absline}{\the\absline@num+1}%
2019 \numdef{\next@abslineR}{\the\absline@numR+1}%
2020 \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
2021 \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
2022 \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
2023 \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
2024 }{}
2025 \IfStrEq{\led@pb@setting}{after}{
2026 \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
2027 \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
2028 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
2029 \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
2030 }{}
2031 \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
2032 \if@pb\pagebreak[4]\fi
2033 }
2034 %

```

**\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).

```

2035 \newcommand*{\columnseparator}{%
2036 \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
2037 \newdimen\columnrulewidth
2038 \columnrulewidth=\z@
2039
2040 %

```

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

```

\columns@position
2041 \newcommand*{\columnspostion}[1]{%
2042 \xdef\columns@position{#1}%
2043 }%
2044 \xdef\columns@position{R}%
2045 %

```

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

```

2046 \newlength{\beforecolumnseparator}%
2047 \setlength{\beforecolumnseparator}{-2pt}%
2048
2049 \newlength{\aftercolumnseparator}%
2050 \setlength{\aftercolumnseparator}{-2pt}%
2051
2052 %

```

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

```

2053 \newcommand{\setwidthliketwocolumns@L}{%
2054 % Temporary dimension, initially equal to the standard hsize, i.e. text
width
2055 % \begin{macrocode}
2056 \newdimen\temp%
2057 \temp=\hsize%
2058 %

```

Hsize : Left + Right width

```

2059 \hsize=\Lcolwidth%
2060 \advance\hsize\Rcolwidth%
2061 %

```

Now, calculating the remaining space

```

2062 \advance\temp-\hsize%
2063 %

```

And multiply the hsize by 2/3 of this space

```

2064 \multiply\temp by 2%
2065 \divide\temp by 3%
2066 \advance\hsize\temp%
2067 }%
2068
2069 \newcommand{\setpositionliketwocolumns@L}{%
2070 \renewcommand{\ledrlfill}{\hfill}%
2071 }%
2072
2073 \newcommand{\setnotespositionliketwocolumns@L}{%
2074 }%
2075
2076
2077 %

```

```

2078 \newcommand{\setwidthliketwocolumns@C}{%
2079 % Temporary dimension, initially equal to the standard hsize, i.e. text
width
2080 %

```

```

2081 \newdimen\temp%
2082 \temp=\hsize%
2083 % Hsize : Left + Right width
2084 %

```

```

2085 \hsize=\Lcolwidth%
2086 \advance\hsize\Rcolwidth%
2087 % Now, calculating the remaining space
2088 %

```

```

2089 \advance\temp-\hsize%
2090 %

```

And multiply the hsize by 1/2 of this space

```

2091 \divide\temp by 2%
2092 \advance\hsize\temp%
2093 }%
2094
2095 \newcommand{\setpositionliketwocolumns@C}{%
2096 \doinsidelinehook{\hfill}%
2097 \renewcommand{\ledrlfill}{\hfill}%
2098 }%
2099
2100 \newcommand{\setnotespositionliketwocolumns@C}{%
2101 \newdimen\temp%
2102 \newdimen\tempa%
2103 \temp=\hsize%
2104 \tempa=\Lcolwidth%
2105 \advance\tempa\Rcolwidth%
2106 \advance\temp-\tempa%
2107 \divide\temp by 2%
2108 \leftskip=\temp%
2109 \rightskip=-\temp%
2110 }%
2111
2112 \newcommand{\setwidthliketwocolumns@R}{%
2113 %

```

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

```

2114 \newdimen\temp%
2115 \temp=\hsize%
2116 %

```

Hsize : Left + Right width

```

2117 \hsize=\Lcolwidth%
2118 \advance\hsize\Rcolwidth%
2119 %

```

Now, calculating the remaining space

```

2120 \advance\temp-\hsize%
2121 %

```

And multiply the hsize by 2/3 of this space

```

2122 \multiply\temp by 2%
2123 \divide\temp by 3%
2124 \advance\hsize\temp%
2125 }%
2126
2127 \newcommand{\setpositionliketwocolumns@R}{%
2128 \doinsidelinehook{\hfill}%
2129 }%
2130
2131 \newcommand{\setnotespositionliketwocolumns@R}{%
2132 \newdimen\temp%
2133 \newdimen\tempa%
2134 \temp=\hsize%
2135 \tempa=\Lcolwidth%
2136 \advance\tempa\Rcolwidth%
2137 \advance\temp-\tempa%
2138 \divide\temp by 2%
2139 \leftskip=\temp%
2140 \rightskip=-\temp%
2141 }%
2142
2143 %

```

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

```

2144 \newcommand{\Columns@print@before@pstart}{%
2145 \ifbool{expr}{%
2146 test{\ifcsstring{before@pstartL@the\l@dpscl}{\at@every@pstart}}}%
2147 and test {\ifcsstring{before@pstartR@the\l@dpscl}{\at@every@pstart}}}%
2148 and test {\ifdefempty{\at@every@pstart}}}%
2149 }%
2150 {%
2151 \hb@xt@ \hsize{%
2152 \ifdefstring{\columns@position}{L}{-}{\hfill }%
2153 \par\parbox[t]{}[t]{\Lcolwidth}{%
2154 \csuse{before@pstartL@the\l@dpscl}%
2155 }%
2156 \print@columnseparator%
2157 \parbox[t]{}[t]{\Rcolwidth}{%
2158 \initnumbering@sectcountR%
2159 \csuse{before@pstartR@the\l@dpscl}%
2160 }%
2161 \ifdefstring{\columns@position}{R}{-}{\hfill}%
2162 }%
2163 }%
2164 \global\csundef{before@pstartL@the\l@dpscl}%
2165 \global\csundef{before@pstartR@the\l@dpscl}%
2166 }%

```

```

2167 \newcommand{\Columns@print@after@pend}{%
2168   \ifboolexpr{%
2169     test{\ifcsstring{after@pendL@the\l@dpscl}{\at@every@pend}}%
2170     and test {\ifcsstring{after@pendR@the\l@dpscR}{\at@every@pend}}%
2171     and test {\ifdefempty{\at@every@pend}}}%
2172   {%
2173     \hb@xt@ \hsize{%
2174       \ifdefstring{\columns@position}{L}{\hfill }%
2175       \parbox[t] [] [t]{\Lcolwidth}{%
2176         \csuse{after@pendL@the\l@dpscl}%
2177       }%
2178       \print@columnseparator%
2179       \parbox[t] [] [t]{\Rcolwidth}{%
2180         \initnumbering@sectcountR%
2181         \csuse{after@pendR@the\l@dpscR}%
2182       }%
2183       \ifdefstring{\columns@position}{R}{\hfill}%
2184     }%
2185   }%
2186 }%
2187 \global\csundef{after@pendL@the\l@dpscl}%
2188 \global\csundef{after@pendR@the\l@dpscR}%
2189 }%
2190 %

```

## 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`

```

2191 \newcount\numpagelinesL
2192 \newcount\numpagelinesR
2193 \newcount\l@dminpagelines
2194
2195 %

```

### 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.

```

2196 \newcommandx*{\Pages}[1][1,usedefault]{%
2197   \ifl@dpairing%

```



```

2198 \led@err@Pages@InsideEnv%
2199 \fi%
2200 \ifstrequal{#1}{mainmatter}{\Pages@mainmattertrue}{\Pages@mainmatterfalse}%
2201 }%
2202 \eledsection@correcting@skip=-2\baselineskip% line correcting for section
2203 titles.
2204 \parledgroup@notespacing@set@correction%
2205 \typeout{}%
2206 \typeout{***** PAGES *****}%
2207 \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else%
2208 \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2209 \fi%
2210 %

```

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

```

2209 \cleartol@evenpage%
2210 \l@dprintingpagetrue%
2211 \begingroup%
2212 %

```

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

```

2213 \setlength{\Lcolwidth}{\textwidth}%
2214 \setlength{\Rcolwidth}{\textwidth}%
2215 %

```

```

2216 \l@dzeropenalties%
2217 \endgraf\global\num@lines=\prevgraf%
2218 \global\num@linesR=\prevgraf%
2219 \global\par@line=\z@%
2220 \global\par@lineR=\z@%
2221 \global\l@dpscL=\z@%
2222 \global\l@dpscR=\z@%
2223 \writtenlinesLfalse%
2224 \writtenlinesRfalse%
2225 %

```

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

```

2226 \let\print@Xnotes\print@Xnotes@forpages%
2227 \let\print@notesX\print@notesX@forpages%
2228 %

```

Check if there are chunks to be processed.

```

2229 \check@pstarts%
2230 \loop\if@pstarts%
2231 %

```

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

```

2232     \global\advance\l@dpscL \@ne%
2233     \global\advance\l@dpscR \@ne%
2234 %

```

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

```

2235     \getlinesfromparlistL%
2236     \getlinesfromparlistR%
2237     \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2238     {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
2239     \check@pstarts%
2240     \repeat%
2241 %

```

Zero the counts again, ready for the next bit.

```

2242     \global\l@dpscL=\z@%
2243     \global\l@dpscR=\z@%
2244 %

```

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

```

2245     \getlinesfrompagelistL%
2246     \getlinesfrompagelistR%
2247     \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2248     {\l@dminpagelines}%
2249 %

```

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.

```

2250     \check@pstarts%
2251     \if@pstarts%
2252 %

```

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

```

2253     \global\advance\l@dpscL \@ne%
2254     \global\advance\l@dpscR \@ne%
2255     \restore@pstartL@pc%
2256     \restore@pstartR@pc%
2257 %

```

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

```

2258     \global\@donereallinesL=\z@%
2259     \global\@donetotallinesL=\z@%
2260     \global\@donereallinesR=\z@%
2261     \global\@donetotallinesR=\z@%
2262 %

```

Start a loop over the boxes (chunks).

```

2263 \checkraw@text%
2264 %
2265 % \beginngroup
2266 { \loop\ifaraw@text%
2267 %

```

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

```

2268 \checkpageL%
2269 \l@duselanguage{\theledlanguageL}%
2270 { \loop\ifl@dsamepage%
2271 %

```

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

```

2272 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2273 \csuse{before@pstartL@the\l@dpscL}%
2274 \global\csundef{before@pstartL@the\l@dpscL}%
2275 \do@lineL%
2276 \xifinlist{the\l@dpscL}{\eled@sections@@}
2277 {\print@eledsectionL}%
2278 {}%
2279 \advance\numpagelinesL \@ne%
2280 %

```

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

```

2281 \ifshiftedpstarts%
2282 \ifdim\ht\l@dleftbox>0pt%
2283 \parledgroup@correction@notespacing{L}%
2284 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2285 \else%
2286 \unless\ifadvancedshiftedpstarts%
2287 \dimen0=\pagetotal%
2288 \advance\dimen0 by \baselineskip%
2289 \global\pagetotal=\dimen0%
2290 \else%
2291 \ifnomaxlines%
2292 \numdef{\@tmp}{the\l@dpscL+1}%
2293 \ifcsdef{minpage@pstart@\@tmp}{%
2294 \ifnumless{the\c@page}{\csuse{
minpage@pstart@\@tmp}}}%
2295 {\dimen0=\pagetotal%
2296 \advance\dimen0 by \baselineskip%

```

```

2297             \global\pagetotal=\dimen0%
2298             }%
2299             {}%
2300             }{}%
2301             \fi%
2302             \fi%
2303             \fi%
2304         \else%
2305             \parledgroup@correction@notespacing{L}%
2306             \hb@xt@ \hsize{\ledstrutL\unhbox\l@leftbox}%
2307             \fi%
2308 %

```

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

```

2309         \get@nextboxL%
2310         \global\l@dskipversenumberfalse%
2311         \ifprint@last@after@pendL%
2312             \csuse{after@pendL@the\l@dpscL}%
2313             \global\csundef{after@pendL@the\l@dpscL}%
2314             \fi%
2315         \checkpageL%
2316         \checkverseL%
2317         \checkpbL%
2318         \repeat%
2319 %

```

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.

```

2320         \ifl@dpagfull%
2321             \@writelinesonpageL{\the\numpagelinesL}%
2322         \else%
2323             \@writelinesonpageL{1000}%
2324         \fi%
2325 %

```

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.

```

2326         \numpagelinesL \z@%
2327         \parledgroup@correction@notespacing@init%
2328         \clearl@leftpage }%
2329 %

```

Now do the same for the right text.

```

2330 \checkpageR%
2331 \l@duselanguage{\theledlanguageR}%
2332 {
2333   \loop\ifl@dsamepage%
2334     \initnumbering@sectcountR%
2335     \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2336     \csuse{before@pstartR@the\l@dpscR}%
2337     \global\csundef{before@pstartR@the\l@dpscR}%
2338     \do@lineR%
2339     \xifinlist{the\l@dpscR}{\eled@sectionsR@@}%
2340     {\print@eledsectionR}%
2341     {}%
2342     \advance\numpagelinesR \@ne%
2343     \ifshiftedpstarts%
2344       \ifdim\ht\l@drightbox>0pt%
2345         \parledgroup@correction@notespacing{R}%
2346         \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2347       \else%
2348         \unless\ifadvancedshiftedpstarts%
2349           \dimen0=\pagetotal%
2350           \advance\dimen0 by \baselineskip%
2351           \global\pagetotal=\dimen0%
2352         \else%
2353           \ifnomaxlines%
2354             \numdef{\@tmp}{\the\l@dpscR+1}%
2355             \ifcsdef{minpage@pstart@\@tmp}{%
minpage@pstart@\@tmp}%
2356               {\dimen0=\pagetotal%
2357                 \advance\dimen0 by \baselineskip%
2358                 \global\pagetotal=\dimen0%
2359                 }%
2360               {}%
2361             }{}%
2362             \fi%
2363             \fi%
2364             \fi%
2365           \else%
2366             \parledgroup@correction@notespacing{R}%
2367             \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2368             \fi%
2369             \get@nextboxR%
2370             \global\l@dskipversenumberRfalse%
2371             \ifprint@last@after@pendR%
2372               \csuse{after@pendR@the\l@dpscR}%
2373               \global\csundef{after@pendR@the\l@dpscR}%
2374             \fi%
2375             \checkpageR%
2376             \checkverseR%
2377             \checkpbR%
2378             \repeat%

```

```

2379 \ifl@dpagfull%
2380 \@writelinesonpageR{\the\numpagelinesR}%
2381 \else%
2382 \@writelinesonpageR{1000}%
2383 \fi%
2384 \numpagelinesR=\z@%
2385 \parledgroup@correction@notespacing@init%
2386 %

```

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

```

2387 \clearl@drightpage}%
2388 %

```

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.

```

2389 \checkraw@text%
2390 \ifaraw@text%
2391 \getlinesfrompagelistL%
2392 \getlinesfrompagelistR%
2393 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2394 \l@dminpagelines}%
2395 \fi%
2396 \repeat}%
2397 %

```

We have now output the text from all the chunks.

```

2398 \fi%
2399 %

```

Make sure that there are no inserts hanging around.

```

2400 \flush@notes%
2401 \flush@notesR%
2402 \endgroup%
2403 %

```

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

```

2404 \global\l@dpscL=\z@%
2405 \global\l@dpscR=\z@%
2406 \global\l@dnumstartsL=\z@%
2407 \global\l@dnumstartsR=\z@%
2408 \global\instanzaLfalse%
2409 \global\instanzaRfalse%
2410 \l@dprintingpagesfalse%
2411 \finish@Pages@notes%Needed to prevent final notes overlap line number
2412 \ignorespaces}
2413
2414
2415 %

```

### 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.

```
2416 \newcommand{\finish@Pages@notes}{%
2417   \def\do##1{%
2418     %
```

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

```
2419   \ifnocritical{%
2420     \global\newnamebox{##1footins}
2421   \fi
2422   \ifnofamiliar{%
2423     \global\newnamebox{footins##1}
2424   \fi
2425 %
```

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

```
2426   \ifvoid\csuse{##1footins}%
2427     \ifvoid\csuse{footins##1}\else%
2428       \newpage\null%
2429       \listbreak%
2430     \fi%
2431   \else%
2432     \newpage\null%
2433     \listbreak%
2434   \fi%
2435 }%
2436 \dolistloop{\@series}%
2437 }%
2438 %
```

### XIX.4 Struts

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

```
\ledstrutR
2439 \newcommand*{\ledstrutL}{}
2440 \newcommand*{\ledstrutR}{}
2441
2442 %
```

### 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@devenpage` is similar except that it first checks to see if it is already on an empty page.

```

2443 \providecommand{\cleartoevenpage}[1][\@empty]{%
2444 \clearpage
2445 \ifodd\c@page\hbox{#1}\clearpage\fi}
2446
2447 \newcommand*\cleartol@devenpage{%
2448 \ifdim\pagetotal<\topskip% on an empty page
2449 \else
2450 \clearpage
2451 \Pages@mainmatter%
2452 \fi
2453 \ifodd\c@page%
2454 \ifprevpgnotnumbered%
2455 \addtocounter{par@page}{-1}%
2456 \ifdef{\prevpgstyle}{\thispagestyle{\prevpgstyle}}{}%
2457 \fi%
2458 \hbox{ }\clearpage%
2459 \fi%
2460 }%
2461 %

```

`\clearl@dleftpage` and `\clearl@drightpage` 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`.

```

2462 \newcommand*\clearl@dleftpage{%
2463 \ifdim\pagetotal=0pt\hbox{ }\fi%
2464 \newpage%
2465 \ifodd\c@page\else
2466 \led@err@LeftOnRightPage
2467 \hbox{ }%
2468 \cleardoublepage
2469 \fi}
2470
2471 \newcommand*\clearl@drightpage{%
2472 \ifdim\pagetotal=0pt\hbox{ }\fi%
2473 \newpage%
2474 \ifodd\c@page
2475 \led@err@RightOnLeftPage
2476 \hbox{ }%
2477 \cleartoevenpage
2478 \fi}
2479
2480 %

```



## XIX.6 Lines managing

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

```

2481 \newcommand*\getlinesfromparlistL{%
2482   \ifx\linesinpar@listL\empty
2483     \gdef\@cs@linesinparL{0}%
2484   \else
2485     \gl@p\linesinpar@listL\to\@cs@linesinparL
2486   \fi}
2487 \newcommand*\getlinesfromparlistR{%
2488   \ifx\linesinpar@listR\empty
2489     \gdef\@cs@linesinparR{0}%
2490   \else
2491     \gl@p\linesinpar@listR\to\@cs@linesinparR
2492   \fi}
2493
2494 %

```

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

```

2495 \newcommand*\getlinesfrompagelistL{%
2496   \ifx\linesonpage@listL\empty
2497     \gdef\@cs@linesonpageL{1000}%
2498   \else
2499     \gl@p\linesonpage@listL\to\@cs@linesonpageL
2500   \fi}
2501 \newcommand*\getlinesfrompagelistR{%
2502   \ifx\linesonpage@listR\empty
2503     \gdef\@cs@linesonpageR{1000}%
2504   \else
2505     \gl@p\linesonpage@listR\to\@cs@linesonpageR
2506   \fi}
2507
2508 %

```

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

```

2509 \newcommand*\@writelinesonpageL[1]{%
2510   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2511   \next}
2512 \newcommand*\@writelinesonpageR[1]{%
2513   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2514   \next}
2515
2516 %

```

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

`\l@dcalc@minoftwo` `\l@dcalc@minoftwo{⟨num⟩}{⟨num⟩}{⟨count⟩}` sets `⟨count⟩` to the minimum of the two `⟨num⟩`.

```

2517 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2518   \ifnum #2>#1\relax
2519     #3=#2\relax
2520   \else
2521     #3=#1\relax
2522   \fi}
2523 \newcommand*{\l@dcalc@minoftwo}[3]{%
2524   \ifnum #2<#1\relax
2525     #3=#2\relax
2526   \else
2527     #3=#1\relax
2528   \fi}
2529 %
2530 %

```

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

```

\ifl@dsamepage \checkpageL
\l@dsamepagetrue \l@dsamepagetrue
\l@dsamepagefalse \ifl@dsamepage
\ifl@dpagfull \l@dsamepagefalse
\l@dpagfulltrue \l@dsamepagetrue
\l@dpagfullfalse \l@dsamepagefalse
\checkpageL
\checkpageR
2531 \newif\ifl@dsamepage
2532 \l@dsamepagetrue
2533 \newif\ifl@dpagfull
2534
2535 \newcommand*{\checkpageL}{%
2536   \l@dpagfulltrue
2537   \l@dsamepagetrue
2538   \check@goal
2539   \ifdim\pagetotal<\ledthegoal
2540     \ifnum\umpagelinesL<\l@dminpagelines
2541       \else
2542         \ifnomaxlines%
2543         \else%
2544           \l@dsamepagefalse%
2545           \l@dpagfullfalse%
2546         \fi%
2547       \fi
2548     \else
2549       \l@dsamepagefalse
2550       \l@dpagfulltrue

```

```

2551 \fi%
2552 \ifprint@last@after@pendL%
2553 \l@dpagfullfalse%
2554 \l@dsamepagefalse%
2555 \print@last@after@pendLfalse%
2556 \fi%
2557 }%
2558
2559 \newcommand*\checkpageR{%
2560 \l@dpagfulltrue
2561 \l@dsamepagetrue
2562 \check@goal
2563 \ifdim\pagetotal<\ledthegoal
2564 \ifnum\numpagelinesR<\l@dminpagelines
2565 \else
2566 \ifnomaxlines%
2567 \else%
2568 \l@dsamepagefalse%
2569 \l@dpagfullfalse%
2570 \fi%
2571 \fi
2572 \else
2573 \l@dsamepagefalse
2574 \l@dpagfulltrue
2575 \fi%
2576 \ifprint@last@after@pendR%
2577 \l@dpagfullfalse%
2578 \l@dsamepagefalse%
2579 \print@last@after@pendRfalse%
2580 \fi%
2581 }%
2582
2583 %

```

**\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.

```

2584 \newcommand{\checkpbL}{
2585 \IfStrEq{\led@pb@setting}{after}{
2586 \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@dpagfulltrue\
l@dsamepagefalse}{
2587 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@dpagfullfalse\
l@dsamepagetrue}{
2588 }{
2589 \IfStrEq{\led@pb@setting}{before}{
2590 \numdef{\next@absline}{\the\absline@num+1}
2591 \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagfulltrue\
l@dsamepagefalse}{
2592 \xifinlistcs{\next@absline}{l@prev@nopb}{\l@dpagfullfalse\
l@dsamepagetrue}{

```

```

2593 }{}
2594 }
2595
2596 \newcommand{\checkpbR}{
2597   \IfStrEq{\led@pb@setting}{after}{
2598     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@pagefulltrue\
2599     l@dsamepagefalse}{}
2600     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@pagefullfalse\
2601     l@dsamepagetrue}{}
2602   }{}
2603   \IfStrEq{\led@pb@setting}{before}{
2604     \numdef{\next@abslineR}{\the\absline@numR+1}
2605     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@pagefulltrue\
2606     l@dsamepagefalse}{}
2607     \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@pagefullfalse\
2608     l@dsamepagetrue}{}
2609   }{}
2610 }
2611 %

```

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

```

2608 \newcommand{\checkverseL}{
2609   \ifinstanzaL
2610     \iflednopbinverse
2611       \ifinserthangingsymbol
2612         \numgdef{\prev@abslineverse}{\the\absline@num-1}
2613         \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{}
2614         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\
2615         prev@abslineverse}\fi}{}
2616       \fi
2617     \fi
2618   }
2619 \newcommand{\checkverseR}{
2620   \ifinstanzaR
2621     \iflednopbinverse
2622       \ifinserthangingsymbolR
2623         \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2624         \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{}
2625         \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\
2626         prev@abslineverse}\fi}{}
2627       \fi
2628     \fi
2629   }
2630 %

```

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

```

2631 \newdimen\ledthegoal
2632 \ifshiftedpstarts
2633     \newcommand*{\@goalfraction}{0.95}
2634 \else
2635     \newcommand*{\@goalfraction}{0.9}
2636 \fi
2637
2638 \newcommand*{\check@goal}{%
2639     \ledthegoal=\@goalfraction\pagegoal}
2640 \newcommand{\setgoalfraction}[1]{%
2641     \xdef\@goalfraction{#1}%
2642 }
2643 %

```

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

```

\ifwrittenlinesL
2644 \newif\ifwrittenlinesL
2645 \newif\ifwrittenlinesR
2646
2647 %

```

## XIX.8 Getting boxes content

`\if@getnextbox` The `\if@getnextbox` boolean is switched to true if we can get the next chunk in a page after finished previous chunk. That is:

- If we use the `nosyncpstarts` option, in any case
- If we do not use it, only when the number or real or blank line of the current chunk is equal or greater to the maximum number of line in the current pair of chunks.

```

2648 \newif\if@getnextbox%
2649 %

```

`\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.

```

2650 \newcommand*{\get@nextboxL}{%
2651     \ifvbox\namebox{1@dLcolrawbox\the\l@dpscl}% box is not empty
2652 %

```

The current box is not empty; do nothing.

```

2653 \else%
2654 %

```

The box is empty. By default, we can get the next box

```
2655 \getnextboxtrue%Should be local, but be cautious
2656 %
```

But not when sufficient lines for this page have been generated (except when we don't do any synchronization whatsoever). output.

```
2657 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscL}>\@donetotallinesL
2658 \parledgroup@notes@endL%
2659 \unless\ifnosyncpstarts%
2660 \getnextboxfalse%
2661 %
```

If we use the nomaxlines option, we will start at new page, but we take count of the lines to be typeset for the actual right chunk on the right page, before starting new chunk on the left page.

```
2662 \ifnomaxlines%
2663 \ifdim\pagetotal<\ledthegoal%
2664 \numdef{\@tmp}{\l@dpscL+1}%
2665 \ifcsdef{afterlines@pstart@\@tmp R}{%
2666 \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp R}}
2667 %
2668 {}%
2669 {\ifcsdef{minpage@pstart@\@tmp}%
2670 {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2671 {\ifnum\numpagelinesL=\l@dminpagelines%
2672 \getnextboxtrue%
2673 \fi%
2674 }%
2675 {\getnextboxtrue}}%
2676 {\getnextboxtrue}%
2677 }%
2678 {}%
2679 \fi%
2680 \fi%
2681 \fi%
2682 \else%
2683 \ifnomaxlines%
2684 \numdef{\@tmp}{\the\l@dpscL+1}%
2685 \ifcsdef{minpage@pstart@\@tmp}{%
2686 \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2687 {\ifdimgreater{\pagetotal}{\ledthegoal}%
2688 {\getnextboxtrue}%
2689 {\getnextboxfalse}%
2690 }%
2691 {\getnextboxtrue}%
2692 }-}%
2693 \fi%
2694 \fi%
```

2695 %

Sufficient lines have been output.

```
2696 \if@getnextbox%
2697 \ifnum\usernamecount{1@dmaxlinesinpar\the\1@dpscl}=\@donetotallinesL
2698 \parledgroup@notes@endL
2699 \fi
2700 \ifwrittenlinesL\else
2701 %
```

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

```
2702 \@writelinesinparL
2703 \writtenlinesLtrue
2704 \fi
2705 \ifnum\1@dnumstartsL>\1@dpscl
2706 %
```

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

```
2707 \writtenlinesLfalse
2708 \ifbypstart@
2709 \global\line@num=0%
2710 \resetprevline@%
2711 \fi
2712 % Add the content of the optional argument of the previous \protect\cs{pend
}.
2713 \begin{macrocode}
2714 \csuse{after@pendL@the\1@dpscl}%
2715 \global\csundef{after@pendL@the\1@dpscl}%
2716 %
```

Check the number of lines

```
2717 \1@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\1@dpscl}}%
2718 {\the\@donetotallinesL}%
2719 {\usernamecount{1@dmaxlinesinpar\the\1@dpscl}}}%
2720 \global\@donetotallinesL \z@
2721 %
```

Go to the next pstart

```
2722 \global\advance\1@dpscl \@ne
2723 \global\pstartnumtrue%
2724 \restore@pstartL@pc%
2725 %
```

Add notes of parallel ledgroup.

```
2726 \parledgroup@notes@endL
2727 \parledgroup@correction@notespadding@final{L}
2728 \else
2729 %
```

```

2730     \print@last@after@pendLtrue%
2731     \fi
2732   \fi
2733 \fi}
2734 %

2735 \newcommand*{\get@nextboxR}{%
2736   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2737   \else% box is empty
2738     \@getnextboxtrue%
2739     \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2740       \parledgroup@notes@endR
2741       \unless\ifnosyncpstarts%
2742         \@getnextboxfalse%
2743       \ifnomaxlines%
2744         \ifdim\pagetotal<\ledthegoal%
2745           \numdef{\@tmp}{\l@dpscR+1}%
2746           \ifcsdef{afterlines@pstart@\@tmp L}{%
2747             \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp L}}
2748             {}%
2749             {\ifcsdef{minpage@pstart@\@tmp}%
2750              {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2751               {\ifnum\numpagelinesR=\l@dminpagelines%
2752                \@getnextboxtrue%
2753               }%
2754             }%
2755             {\@getnextboxtrue}}%
2756           {\@getnextboxtrue}%
2757         }%
2758       }%
2759     }%
2760   \fi%
2761 \fi%
2762 \fi%
2763 \else%
2764   \ifnomaxlines%
2765     \numdef{\@tmp}{\the\l@dpscR+1}%
2766     \ifcsdef{minpage@pstart@\@tmp}{%
2767       \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2768       {\ifdimgreater{\pagetotal}{\ledthegoal}%
2769        {\@getnextboxtrue}%
2770        {\@getnextboxfalse}%
2771       }%
2772       {\@getnextboxtrue}%
2773     }{}
2774   \fi%
2775 \fi%
2776 \if@getnextbox%
2777   \ifnum\usenamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR

```



```

2778 \parledgroup@notes@endR
2779 \fi
2780 \ifwrittenlinesR\else
2781 \@writelinesinparR
2782 \writtenlinesRtrue
2783 \fi
2784 \ifnum\l@dnumpstartsR>\l@dpscR
2785 \writtenlinesRfalse
2786 \ifbypstartR
2787 \global\line@numR=0%
2788 \resetprevline@%
2789 \fi
2790 \csuse{after@pendR@the\l@dpscR}%
2791 \global\csundef{after@pendR@the\l@dpscR}%
2792 \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
2793 {\the\@donetotallinesR}%
2794 {\usernamecount{l@dmaxlinesinpar\the\l@dpscR}}%
2795 \global\@donetotallinesR \z@
2796 \global\advance\l@dpscR \@ne
2797 \global\pstartnumRtrue%
2798 \restore@pstartR@pc%
2799 \parledgroup@notes@endR
2800 \parledgroup@correction@notes@spacing@final{R}
2801 \else
2802 \print@last@after@pendRtrue%
2803 \fi
2804 \fi
2805 \fi}
2806
2807 %

```

## XX Page numbering

### XX.1 Global options

The `sameparallelpagenumber` option allows the same page number on both left and right side. The `prevpgnotnumbered` 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

`\par@patch@pagenumbering`

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 `\pagenumbers` macro `\pagenumbers`, which is called by some `\xxxmatter` commands. In the case of `memoir` class using, we insert it at the end of `\@mempnum`. When using `\pagenumbers`, we also need to restart `par@page` counter. Consequently, we have wrapped `\par@patch@thepage` and counter restart in `\par@patch@pagenumbers`. We also call `\par@patch@thepage` it at the beginning of the document.

```

2808
2809 \newcommand{\par@patch@thepage}{%
2810   \ifboolexpr{%
2811     bool{sameparallelpagenumbers}%
2812     or bool{prevpgnotnumbered}%
2813   }%
2814   {%
2815     \patchcmd{\thepage}%
2816       {page}{par@page}%
2817       {}%
2818       {\led@error@fail@patch@thepage}%
2819   }{}%
2820 }%
2821
2822 \newcommand{\par@patch@pagenumbers}{%
2823   \ifboolexpr{%
2824     bool{sameparallelpagenumbers}%
2825     or bool{prevpgnotnumbered}%
2826   }%
2827   {%
2828     \setcounter{par@page}{1}%
2829   }%
2830   {}%
2831   \par@patch@thepage%
2832 }%
2833
2834 \ifl@dmemoir%
2835   \apptocmd{\@mempnum}%
2836     {\par@patch@pagenumbers}%
2837     {}%
2838     {\led@error@fail@patch@@mempnum}%
2839
2840 \else%
2841   \apptocmd{\pagenumbers}%
2842     {\par@patch@pagenumbers}%
2843     {}%
2844     {\led@error@fail@patch@pagenumbers}%
2845 \fi%
2846
2847 \AtBeginDocument{\par@patch@thepage}%
2848 %

```

`\@outputpage` As its name says, `\@outputpage` is a  $\TeX$ '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.

```

2849 \AtBeginDocument{%
2850   \apptocmd{\@outputpage}{%
2851     \ifsameparallelpagelapnumber%
2852     \ifl@dprintingpages%
2853     \ifodd\c@page\else%
2854       \stepcounter{par@page}%
2855     \fi%
2856   \else%
2857     \stepcounter{par@page}%
2858   \fi%
2859 \else%
2860   \stepcounter{par@page}%
2861 \fi%
2862 }%
2863 {}%
2864 {\led@error@fail@patch@\@outputpage}%
2865 }
2866 %

```

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

```

2867 \newcounter{par@page}%
2868 \setcounter{par@page}{1}%
2869 %

```

## XX.2 mainmatter option of \Pages

The optional argument of `\Pages` could be equal to `mainmatter`. In this case the boolean `\ifPages@mainmatter` is set to true, and some special things are done in `\Pages@mainmatter`, called by `\cleartol@evenpage`.

```

\ifPages@mainmatter 2870 \newif\ifPages@mainmatter
\Pages@mainmatter 2871 \newcommand{\Pages@mainmatter}{%
2872   \ifPages@mainmatter%
2873     \pagenumbering{arabic}%
2874     \addtocounter{page}{1}%
2875     \addtocounter{par@page}{-1}%
2876     \patchcmd{\thepage}{page}{par@page}{}{}%
2877   \fi%
2878 }
2879 %

```

## 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 misaligned.

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

```
\chapter 2880 \pretocmd{\chapter}{%
2881   \ifl@dprintingpages%
2882   \vbox{}%
2883   \fi%
2884 }%
2885 {}%
2886 {}%
2887 %
```

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

```
2888 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2889 \eledsectnotoc{R}
2890 %
```

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

```
2891 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2892 \eledsectmark{L}
2893 %
```

`\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`.

```
2894 \newskip\eledsection@correcting@skip
2895 %
```

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

```
2896 \newwrite\eled@sectioningR@out
2897 %
```

## 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).

```
2898 \def\l@prev@pbR{}
2899 \def\l@prev@nopbR{}
2900 %
```

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

```
2901 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2902 \newcommand{\ledpbnR}[1]{\write\linenum@outR{\string\led@pbnR{#1}}}
2903 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2904 \newcommand{\lednopbnR}[1]{\write\linenum@outR{\string\led@nopbnR{#1}}}
2905 %
```

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

```
2906 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2907 \newcommand{\led@pbnR}[1]{\listxadd{\l@prev@pbR}{#1}}
2908 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2909 \newcommand{\led@nopbnR}[1]{\listxadd{\l@prev@nopbR}{#1}}
2910 %
```

## 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).

```
2911 \newmarks\parledgroup@
2912 \newmarks\parledgroup@series
2913 \newmarks\parledgroup@type
2914 %
```

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

```
2915 \newcommand{\parledgroup@notes@startL}{%
2916 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscL}>0%
2917 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{}}%
```

```

2918 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
      bhookXnote@\splitfirstmarks\parledgroup@series}}{ }%
2919 \fi%
2920 \global\ledgroupnotesL@true%
2921 \insert@noterule@ledgroup{L}%
2922 }
2923 \newcommand{\parledgroup@notes@startR}{%
      \ifnum\usernamecount{1@dmxlinesinpar\the\l@dpscR}>0%
2924 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
      bhooknoteX@\splitfirstmarks\parledgroup@series}}{ }%
2925 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
      bhookXnote@\splitfirstmarks\parledgroup@series}}{ }%
2926 \fi%
2927 \global\ledgroupnotesR@true%
2928 \insert@noterule@ledgroup{R}%
2929 }
2930 %
2931 %

```

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

```

2932 \newcommand{\parledgroup@notes@endL}{%
      \global\ledgroupnotesL@false%
2933 }
2934 }
2935 \newcommand{\parledgroup@notes@endR}{%
      \global\ledgroupnotesR@false%
2936 }
2937 }
2938 %

```

`\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`

```

2939 \newcommand{\insert@noterule@ledgroup}[1]{
2940 \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2941 \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2942 \csuse{ifledgroupnotes#1@}
2943 \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2944 \csuse{\splitbotmarks\parledgroup@series footnoterule}
2945 \fi
2946 }
2947 {}
2948 \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2949 \csuse{ifledgroupnotes#1@}
2950 \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2951 \csuse{footnoterule\splitbotmarks\parledgroup@series}
2952 \fi
2953 }{ }
2954 }
2955 {}
2956 }

```

2957 %

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

2958 `\newcommand{\setparledgroupnotespacing}[1]{\gdef\@parledgroupnotespacing`  
`{#1}}`

2959 `\newcommand{\@parledgroupnotespacing}{}`

2960 %

`up@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`.

2961 `\dimdef{\parledgroup@notespacing@correction}{0pt}`

2962 `\newcommand{\parledgroup@notespacing@set@correction}{%`

2963 `{\@getfirstseries\csuse{Xnotefontsize@}\@firstseries}%We suppose all the`  
`series has the same footnote size setup`

2964 `\@parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%`

2965 `\dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing`  
`}%`

2966 `}`

2967 %

`rection@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.

2968 `\newcommand{\parledgroup@correction@notespacing@init}{`

2969 `\dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}`

2970 `\dimdef{\parledgroup@notespacing@correction@modulo}{0pt}`

2971 `}`

2972 `\parledgroup@correction@notespacing@init`

2973 %

`rection@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`.

2974 `\newcommand{\parledgroup@correction@notespacing@final}[1]{`

2975 `\ifparledgroup`

2976 `\vspace{\parledgroup@notespacing@correction@accumulated}`

2977 `\parledgroup@correction@notespacing@init%`

2978 `\ifstrequal{#1}{L}{`

2979 `\numdef{\@checking}{\the\l@dpsscL-1}`

2980 `{`

2981 `\numdef{\@checking}{\the\l@dpsscR-1}`

2982 `}`

```

2983 \dimdef{\@beforenotes@current@diff}{\csuse{@parledgroup@beforenotes@
@checking L}-\csuse{@parledgroup@beforenotes@@checking R}}}%
2984 \ifstrequal{#1}{L}%
2985   {% Left
2986     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\
@beforenotes@current@diff}}}%
2987   }%
2988   {% Right
2989     \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\
@beforenotes@current@diff}}{}
2990   }%
2991 \fi
2992 }
2993 %

```

`\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.

```

2994 \newcommand{\parledgroup@correction@notespacing}[1]{%
2995   \csuse{ifledgroupnotes#1@}%
2996   \vspace{-\parledgroup@notespacing@correction}%
2997   \dimdef{\parledgroup@notespacing@correction@accumulated}{\
parledgroup@notespacing@correction@accumulated+\
parledgroup@notespacing@correction}%
2998   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo+\
parledgroup@notespacing@correction}%
2999   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip
}{\advance\numpagelinesL -\@ne%
3000   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo-\baselineskip}%
3001   }% mean greater than equal
3002   \fi%
3003 }
3004 %

```

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



```

3005 \dimdef\parledgroup@beforenotesL{0pt}
3006 \dimdef\parledgroup@beforenotesR{0pt}
3007 %

```

`\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.

```

3008 \newcommand{\parledgroup@beforenotes@save}[1]{
3009   \ifparledgroup
3010     \csdimgdef{@parledgroup@beforenotes@the\csuse{lednumpstarts#1}#1}{\
csuse{parledgroup@beforenotes#1}}
3011     \csdimgdef{parledgroup@beforenotes#1}{0pt}
3012   \fi
3013 }
3014 %

```

## XXIV Compatibility with eledmac

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

```

3015 \ifeledmaccompat%
3016
3017
3018   \unless\ifnocritical@
3019     \let\onlyXside\Xonlyside
3020   \fi
3021 \fi
3022 %

```

## 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 reledpar 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.

## Appendix A.5 Migration to reledpar 2.4.0

We have fixed a bug which which misaligned left and right sides when a line contained a dotted letter.

We have tested and saw no problem with this correction, but if you see a difference in alignment between version 2.3.0 and 2.4.0, please contact us.

## Appendix A.6 Migration to reledpar 2.5.0

If you use `\stanza` or *astanza* environment, please read Appendix A.12 p. 297.

## 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	
\@adv	.....	1
\@astanza@line	.....	1
\@cs@linesinparL	.....	1
\@cs@linesinparR	.....	1
\@cs@linesonpageL	.....	1
\@cs@linesonpageR	.....	1
\@donereallinesL	.....	1
\@donereallinesR	.....	1
\@donetotallinesL	.....	1
\@donetotallinesR	.....	1
\@eledsectionL	.....	1
\@eledsectionR	.....	1
\@lab	.....	1
\@lopL	.....	1
\@lopR	.....	1
\@nl	.....	1
\@nl@regR	.....	1
\@outputpage	.....	1
\@par@sync@option	.....	1
\@par@this@sync@option	.....	1
\@parledgroupnotespacing	.....	1
\@pend	.....	1
\@pendR	.....	1
\@pstart	.....	1
\@pstartR	.....	1
\@pstartsfalse	.....	1
\@pstartstrue	.....	1
\@ref	.....	1
\@ref@regR	.....	1
\@set	.....	1
\@stopastanza	.....	1
\@writelinesinparL	.....	1
\@writelinesinparR	.....	1
\@writelinesonpageL	.....	1
\@writelinesonpageR	.....	1
\@writepageofparL	.....	1
\@writepageofparR	.....	1
CLASSmemoir	.....	106
COMMAND\@Rlineflag	.....	65, 70
COMMAND\@adv	.....	35, 139
COMMAND\@cs@linesinparL	.....	97

COMMAND\@cs@linesonpageL	97
COMMAND\@eledsectionL	80
COMMAND\@eledsectionR	80
COMMAND\@eledsectmark	108
COMMAND\@eledsectnotoc	108
COMMAND\@footnotetext	72
COMMAND\@goalfraction	12, 101
COMMAND\@l@tempcnta	64
COMMAND\@lab	34, 70, 139
COMMAND\@lopL	39, 91, 97
COMMAND\@lopR	39, 97
COMMAND\@mempnum	106
COMMAND\@namedef	74
COMMAND\@namuse	74
COMMAND\@nl	34, 41, 70, 139
COMMAND\@nl@regR	34
COMMAND\@outputpage	105, 107
COMMAND\@page	70
COMMAND\@par@sync@option	33
COMMAND\@parledgroupnotespacing	111
COMMAND\@pend	39
COMMAND\@pendR	39
COMMAND\@pstart	39
COMMAND\@pstartstrue	79
COMMAND\@ref	37, 38, 41, 139
COMMAND\@ref@regR	37
COMMAND\@set	35, 139
COMMAND\@sw	38
COMMAND\AtBeginPairs	8, 45, 137
COMMAND\AtEveryPend	137–139
COMMAND\AtEveryPstart	2, 16, 52, 81, 137–139
COMMAND\AtEveryPstartCall	2, 16, 52, 138
COMMAND\Clear the right lines for \read@linelist	33
COMMAND\Columns	7, 8, 16, 44, 80, 84, 134–137, 139, 140
COMMAND\Columns@print@after@pend	87
COMMAND\Columns@print@before@pstart	87
COMMAND\Lcolwidth	8, 10, 89
COMMAND\Leftsidehook	134
COMMAND\Leftsidehookend	134
COMMAND\Pages	4, 7, 9, 10, 12, 16, 44, 62, 66, 67, 88, 89, 95, 105, 107, 111, 134, 137–140
COMMAND\Pages@mainmatter	107
COMMAND\Rcolwidth	8, 10, 89
COMMAND\Rightsidehook	134
COMMAND\Rightsidehookend	134
COMMAND\Rlineflag	114
COMMAND\Xmaxhnotes	13
COMMAND\Xnoteswidthliketwocolumns	9, 137
COMMAND\Xonlyside	13, 66, 115
COMMAND\&	17
COMMAND\absline@numR	32

COMMAND\add@penalties	64
COMMAND\add@penaltiesL	64
COMMAND\advanceline	35, 41, 139
COMMAND\affixline@num	59
COMMAND\affixline@numR	59, 134, 135
COMMAND\affixpstart@num	62
COMMAND\affixpstart@numR	62
COMMAND\affixside@note	71
COMMAND\aftercolumnseparator	9, 84, 136
COMMAND\araw@textfalse	79
COMMAND\araw@texttrue	79
COMMAND\at@begin@pairs	45
COMMAND\autopar	16
COMMAND\ballast@count	64
COMMAND\baselineskip	84, 112
COMMAND\bbl@set@language	75, 76, 139
COMMAND\beforecolumnseparator	9, 84, 136
COMMAND\begin	17
COMMAND\beginnumbering	14–16, 25, 34, 85, 135, 136, 139
COMMAND\beginnumberingR	40
COMMAND\bf	135
COMMAND\bfseries	135
COMMAND\brokenpenalty	64
COMMAND\chapter	108, 134
COMMAND\check@goal	101
COMMAND\check@pstarts	79
COMMAND\checkpageL	91, 98
COMMAND\checkpb@columns	83
COMMAND\checkpbL	99
COMMAND\checkpbR	99
COMMAND\checkraw@text	79
COMMAND\checkverseL	100
COMMAND\checkverseR	100
COMMAND\clear(double)page	95
COMMAND\clearl@dleftpage	96
COMMAND\clearl@drightpage	96
COMMAND\clearpage	96, 138
COMMAND\cleartoevenpage	95
COMMAND\cleartol@devenpage	95, 107
COMMAND\columnrulewidth	8, 84
COMMAND\columns@position	84
COMMAND\columnseparator	8
COMMAND\columnspan	9, 136
COMMAND\correct@Xfootins@box	67, 68, 138
COMMAND\correct@footinsX@box	67, 138
COMMAND\critext	138
COMMAND\csname	44
COMMAND\displaywidowpenalty	64
COMMAND\do@actions	57
COMMAND\do@actions@fixedcode	134

COMMAND\do@actions@nextR	58
COMMAND\do@actionsR	58, 134
COMMAND\do@ballast	64
COMMAND\do@ballastR	57
COMMAND\do@insidelineLhook	136
COMMAND\do@insidelineRhook	136
COMMAND\do@line	52
COMMAND\do@line(L/R)	55
COMMAND\do@lineL	53, 64, 134, 135
COMMAND\do@lineLhook	134
COMMAND\do@lineR	55, 134–136
COMMAND\do@lineRhook	134
COMMAND\do@lockoff	140
COMMAND\do@lockoffR	36
COMMAND\do@lockon	139
COMMAND\do@lockonR	36
COMMAND\doinsidelineLhook	137
COMMAND\doinsidelineRhook	137
COMMAND\dolineLhook	137
COMMAND\dolineRhook	137
COMMAND\edindex	138
COMMAND\edlabel	135, 138
COMMAND\edtext	37, 41, 137, 138
COMMAND\eled@sectioningR@out	108
COMMAND\eledchapter	138
COMMAND\eledsection	137, 138, 140
COMMAND\eledsection@correcting@skip	108
COMMAND\eledsectmark	19, 108
COMMAND\eledsectnotoc	19, 108
COMMAND\eledxxx	137
COMMAND\end	17
COMMAND\endgraf	51
COMMAND\endlock	41, 139
COMMAND\endnumbering	14, 16, 26, 139
COMMAND\endsub	41, 139
COMMAND\endumbering	14
COMMAND\expandafter	43
COMMAND\extensionchars	25
COMMAND\firstlinenum	15, 136, 140
COMMAND\firstsublinenum	136, 140
COMMAND\fix@page	35, 139
COMMAND\flag@end	41, 134, 137
COMMAND\flag@start	41, 137
COMMAND\flush@notesR	65
COMMAND\footnoteX	42
COMMAND\footnoteXmk	13
COMMAND\footnoteXnomk	13, 42
COMMAND\frontmatter	12, 19
COMMAND\get@nextboxL	135
COMMAND\get@nextboxR	135

COMMAND\getline@numL	56
COMMAND\getline@numR	56
COMMAND\getlinesfrompagelistL	97
COMMAND\getlinesfrompagelistR	97
COMMAND\getlinesfromparlistL	97
COMMAND\getlinesfromparlistR	97
COMMAND\gl@p	43
COMMAND\goalfraction	114
COMMAND\hangingsymbol	114, 135
COMMAND\hfill	83, 84
COMMAND\hidenumbering	15, 139
COMMAND\if@getnextbox	101
COMMAND\ifPages@mainmatter	107
COMMAND\ifbypage@	139
COMMAND\ifbypstart@R	139
COMMAND\ifdim	83
COMMAND\ifinserthangingsymbol	72
COMMAND\ifinserthangingsymbolR	72
COMMAND\ifl@dpagefull	98
COMMAND\ifl@dpageing	22, 137
COMMAND\ifl@dpairing	22, 134
COMMAND\ifl@dsamelang	136
COMMAND\ifl@dsamepage	98
COMMAND\ifl@pagefull	91
COMMAND\ifledRcol	22
COMMAND\iflledRcol	135
COMMAND\ifnumberedpar@	48
COMMAND\ifnumberingR	135
COMMAND\ifnumberpstart	44
COMMAND\ifpst@rtedL	25, 26, 49, 134
COMMAND\ifpst@rtedR	25
COMMAND\ifsublines@	35
COMMAND\insert@countR	37
COMMAND\insert@noterule@ledgroup	110
COMMAND\insertlines@list	37
COMMAND\insertlines@listR	37
COMMAND\inserts@list	48
COMMAND\inserts@listR	63
COMMAND\l@d@nums	41, 65
COMMAND\l@d@set	35, 41, 139
COMMAND\l@dLcolrawbox	48
COMMAND\l@dLcolrawbox1	78
COMMAND\l@dLcolrawbox2	78
COMMAND\l@dRcolrawbox	48
COMMAND\l@dbfnote	72, 139
COMMAND\l@dcalc@maxoftwo	98
COMMAND\l@dcalc@minoftwo	98
COMMAND\l@dchecklang	134, 137
COMMAND\l@dcsnote	136
COMMAND\l@dleftbox	52, 91, 138



COMMAND\l@dlinenumR	32, 134
COMMAND\l@dlsnote	136
COMMAND\l@dmake@labels	70
COMMAND\l@dmaxlinesinpar	90
COMMAND\l@dmaxlinesinpar1	78
COMMAND\l@dminpagelines	90, 134
COMMAND\l@dnumpsstartsL	77, 134
COMMAND\l@dprintingcolumnstrue	138
COMMAND\l@dprintingpagestrue	138
COMMAND\l@dpscL	81, 89, 90, 103
COMMAND\l@dpscR	81, 89, 90
COMMAND\l@drsnote	136
COMMAND\l@dsetupmaxlinecounts	78
COMMAND\l@duselanguage	76, 134
COMMAND\l@dzeromaxlinecounts	78
COMMAND\l@prev@nopbR	109
COMMAND\l@prev@pbR	109
COMMAND\labelstarttrue	135
COMMAND\labelref@list	70
COMMAND\labelref@listR	70
COMMAND\lang	75
COMMAND\last@page@numR	35
COMMAND\led	135
COMMAND\led@nopbR	109
COMMAND\led@nopbnumR	109
COMMAND\led@pbR	109
COMMAND\led@pbnumR	109
COMMAND\ledinnerrote	18
COMMAND\ledleftnote	18
COMMAND\lednopb	18, 83, 99
COMMAND\lednopbR	109
COMMAND\lednopbnumR	109
COMMAND\ledouterote	18
COMMAND\ledpb	83, 99
COMMAND\ledpbR	109
COMMAND\ledpbnumR	109
COMMAND\ledrightnote	18
COMMAND\ledsidenote	18
COMMAND\ledstrutL	134
COMMAND\ledstrutR	134, 140
COMMAND\ledthegoal	101
COMMAND\ledtrutL	134, 140
COMMAND\leftlinenumR	32, 134
COMMAND\let	43
COMMAND\line@list@R	38
COMMAND\line@list@stuff	34, 40
COMMAND\line@margin	29
COMMAND\line@marginR	29, 134
COMMAND\line@numR	32
COMMAND\lineation	15, 138

COMMAND\lineation*	15, 29, 137
COMMAND\lineationR	15, 28, 138
COMMAND\linenum@out	70
COMMAND\linenum@outR	40
COMMAND\linenumberstyle	15, 115
COMMAND\linenumberstyle*	115
COMMAND\linenumberstyleR	15, 115
COMMAND\linenumincrement	15, 136, 140
COMMAND\linenummargin	16, 29, 134, 139
COMMAND\linenummargin*	16, 29, 140
COMMAND\linenummarginR	16, 29, 140
COMMAND\linenumrepR	31, 134
COMMAND\linesinpar@listL	39, 97
COMMAND\linesonpage@listL	39, 97
COMMAND\lock@off	36
COMMAND\lock@on	36
COMMAND\mainmatter	2, 12, 19, 140
COMMAND\makeatletter	55
COMMAND\maxchunks	7, 17, 78
COMMAND\maxhnotesX	13
COMMAND\memorydump	14, 27
COMMAND\n@num	138
COMMAND\new@lineL	41
COMMAND\new@lineR	41
COMMAND\newhookcommand@series	44
COMMAND\newif	138
COMMAND\newpage	95, 96, 138
COMMAND\newseries	44
COMMAND\newseries@	42
COMMAND\newseries@par	42–44
COMMAND\noeledxxx	137
COMMAND\nomark@	42
COMMAND\nomaxlines	39
COMMAND\normalbfnoteX	134, 139
COMMAND\notesXwidthliketwocolumns	9, 137
COMMAND\num@lines	64
COMMAND\num@lines(R)	48
COMMAND\numberingR	26
COMMAND\numberlinefalse	7
COMMAND\numberonlyfirstinline	135
COMMAND\numberpstartfalse	15
COMMAND\numberpstarttrue	15, 135, 140
COMMAND\one@line	48, 72
COMMAND\one@lineR	48
COMMAND\onlyXside	115
COMMAND\onlysideX	13, 66, 139
COMMAND\otherlanguage	140
COMMAND\page@action	35, 139
COMMAND\pagenumbering	106, 140
COMMAND\pages	12

COMMAND\pagetotal	91, 138
COMMAND\par@line	64
COMMAND\par@line(R)	48
COMMAND\par@patch@pagenumbering	106
COMMAND\par@patch@thepage	105, 106
COMMAND\par@sync@option	21
COMMAND\parledgroup@	109
COMMAND\parledgroup@beforenotes@save	113
COMMAND\parledgroup@beforenotesL	112
COMMAND\parledgroup@beforenotesR	112
COMMAND\parledgroup@correction@notespacing	112
COMMAND\parledgroup@correction@notespacing@final	111
COMMAND\parledgroup@correction@notespacing@init	111
COMMAND\parledgroup@notes@endL	110
COMMAND\parledgroup@notes@endR	110
COMMAND\parledgroup@notes@startL	109
COMMAND\parledgroup@notes@startR	109
COMMAND\parledgroup@notespacing@correction	111, 112
COMMAND\parledgroup@notespacing@correction@accumulated	112
COMMAND\parledgroup@notespacing@correction@modulo	112
COMMAND\parledgroup@notespacing@set@correction	111
COMMAND\parledgroup@series	109
COMMAND\parledgroup@type	109
COMMAND\parledgroupnotespacing	114
COMMAND\parledgrouptrue	18
COMMAND\patchcmd	139
COMMAND\pausenumbering	27
COMMAND\pend	3, 7, 10, 16–18, 44, 48, 51, 52, 77, 87, 136, 137, 139, 140
COMMAND\pendL	136, 137
COMMAND\pendR	137
COMMAND\pends	16
COMMAND\prev@nopbR	109
COMMAND\prev@pbR	109
COMMAND\prevpgstyle	22
COMMAND\print@Xnotes	66
COMMAND\print@Xnotes@forpages	66, 138
COMMAND\print@columnseparator	83, 137
COMMAND\print@eledsectionL	54
COMMAND\print@line	54
COMMAND\print@lineL	54
COMMAND\print@notesX@forpages	138
COMMAND\printlines	65
COMMAND\printlinesR	65, 134
COMMAND\pstart	3, 7, 10, 15–18, 28, 41, 44, 48, 49, 51, 52, 77, 81, 87, 135, 136, 139, 140
COMMAND\pstartL	52, 136
COMMAND\pstartR	52, 135, 136
COMMAND\pstartinfootnote	138
COMMAND\rw@text	77
COMMAND\read@linelist	33, 34, 117, 139
COMMAND\ref@reg	37

COMMAND\ref@regR	37, 139
COMMAND\relax	115
COMMAND\reledmac	139
COMMAND\renewcommand	114
COMMAND\resumenumbering	27, 136
COMMAND\resumenumberingR	137
COMMAND\rightlinenumR	32, 134
COMMAND\section	134
COMMAND\section@num	25
COMMAND\selectlanguage	17, 75, 76
COMMAND\set@line	41, 139
COMMAND\set@line@action	35, 139
COMMAND\setRlineflag	16, 114
COMMAND\setgoalfraction	12, 114
COMMAND\sethangingsymbol	17
COMMAND\setline	35, 41, 139
COMMAND\setlinenum	35, 41, 139
COMMAND\setnoteposition...	85
COMMAND\setparledgroupnotespacing	114, 140
COMMAND\setposition...	85
COMMAND\setprintlines	134
COMMAND\setstanzaindents	9, 17, 114
COMMAND\setwidth...	85
COMMAND\sidenotemargin	18, 137
COMMAND\sidenotemargin*	18, 137
COMMAND\skipnumbering	15, 138
COMMAND\sloppy	8
COMMAND\stanza	7, 9, 15, 17, 46, 73, 115, 135
COMMAND\stanzanumtrue	18
COMMAND\startlock	41, 139
COMMAND\startsub	41, 139
COMMAND\sub@action	35, 140
COMMAND\sub@off	70
COMMAND\sub@on	70
COMMAND\subline@numR	32
COMMAND\sublinenumberstyle	16, 115
COMMAND\sublinenumberstyle*	115
COMMAND\sublinenumberstyleR	16, 115
COMMAND\sublinenumincrement	136, 140
COMMAND\sublinenumrepR	31, 134
COMMAND\sza@0@	17
COMMAND\textheight	13
COMMAND\textwidth	46
COMMAND\thefootnoteX	136
COMMAND\theledlanguageL	76
COMMAND\theledlanguageR	76
COMMAND\thepage	19, 105, 106
COMMAND\thepstartL	15, 135
COMMAND\thepstartR	15, 135
COMMAND\thestanzaL	18

COMMAND\thestanzaR	18
COMMAND\vbox	49
COMMAND\vl@dbfnote	72
COMMAND\vskip	110
COMMAND\vsplit	64
COMMAND\widthliketwocolumns	9
COMMAND\widthliketwocolumnsfalse	9
COMMAND\widthliketwocolumnstrue	9
COMMAND\xright@appenditem	43
COMMAND\xspace	20
COMMAND\xxxfootstart	85
COMMAND\xxxmatter	106
ENVIRONMENTLeftside	46
ENVIRONMENTRightside	47
ENVIRONMENTastanza	17, 73, 115, 140
ENVIRONMENTledgroup	6
ENVIRONMENTleft	15
ENVIRONMENTpages	45
ENVIRONMENTpairs	45
PACKAGEEDMAC	116
PACKAGEEDSTANZA	116
PACKAGEEledmac	42, 65, 138
PACKAGEEledpar	138
PACKAGETABMAC	116
PACKAGEbabel	17, 75–77
PACKAGEedmac	115
PACKAGEeledmac	4, 77, 113, 114, 116, 136, 137, 139
PACKAGEeledpar	5, 13, 30, 114, 115, 136–138
PACKAGEetoolbox	83, 109
PACKAGEledmac	5
PACKAGEledpar	1, 5
PACKAGEMemoir	115
PACKAGEMusixtex	136
PACKAGEpolyglossia	17, 75–77
PACKAGEreledmac	1, 3, 5–7, 9, 12, 13, 15–20, 22, 25, 26, 29, 30, 32, 34–37, 39–42, 44, 54, 61, 70, 72, 89, 95, 114, 115, 139, 140
PACKAGEreledpar	1, 3, 5–7, 9–12, 17–22, 28, 32, 33, 39, 42, 44, 70, 114, 115, 139
PACKAGEsetspace	2, 19
PACKAGExkeyval	20

## 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

<code>\affixpstart@numL</code> .....	1
<code>\affixpstart@numR</code> .....	1
<code>\affixside@noteR</code> .....	1
<code>\aftercolumnseparator</code> .....	1, 9
<code>\araw@textfalse</code> .....	1
<code>\araw@texttrue</code> .....	1
<code>astanza (environment)</code> .....	17
<code>\AtBeginPairs</code> .....	1, 8
<code>\AtEveryPstartCall</code> .....	1
<code>\autopar</code> .....	16

**B**

<code>\bbl@set@language</code> .....	1
<code>\beforecolumnseparator</code> .....	1, 9
<code>\beginnumbering</code> .....	14
<code>\beginnumberingR</code> .....	1

**C**

<code>\c@firstlinenumR</code> .....	1
<code>\c@firstsublinenumR</code> .....	1
<code>\c@linenumincrementR</code> .....	1
<code>\c@sublinenumincrementR</code> .....	1
<code>\ch@ck@l@ckR</code> .....	1
<code>\ch@cksub@l@ckR</code> .....	1
<code>\chapter</code> .....	1
<code>\chapterinpages</code> .....	1
<code>\check@goal</code> .....	1
<code>\check@pstarts</code> .....	1
<code>\checkpageL</code> .....	1
<code>\checkpageR</code> .....	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, 8
<code>\Columns</code> .....	1, 8
<code>\columns@position</code> .....	1
<code>\Columns@print@after@pend</code> .....	1
<code>\Columns@print@before@pstart</code> .....	1
<code>\columnseparator</code> .....	1, 8
<code>\columnsposition</code> .....	1, 9
<code>\correct@footinsX@box</code> .....	1
<code>\correct@Xfootins@box</code> .....	1
<code>\countLline</code> .....	1

\countRline .....	1
\critext .....	1

## D

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

## E

\edlabel .....	1
\edtext .....	1
\eled@sectioningR@out .....	1
\eledsection@correcting@skip .....	1
\eledsectmark .....	1, 19
\eledsectnotoc .....	1, 19
\endlock .....	1
\endnumbering .....	1, 14
\endnumberingR .....	1
\endsub .....	1
environments:	
astanza .....	17
Leftside .....	14
pages .....	9
pairs .....	8
Rightside .....	14

## F

\f@x@l@cksR .....	1
\finish@Pages@notes .....	1
\first@linenum@out@Rfalse .....	1
\first@linenum@out@Rtrue .....	1
\firstlinenum .....	1, 15
\firstlinenum* .....	1, 15

<code>\firstlinenumR</code> .....	1, 15
<code>\firstsublinenum</code> .....	1, 15
<code>\firstsublinenum*</code> .....	1, 15
<code>\firstsublinenumR</code> .....	1, 15
<code>\fix@page</code> .....	1
<code>\flag@end</code> .....	1
<code>\flag@start</code> .....	1
<code>\flush@notesR</code> .....	1
<code>\footnoteXmk</code> .....	13
<code>\footnoteXnomk</code> .....	13

## 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> .....	15
-----------------------------------	----

## I

<code>\if@getnextbox</code> .....	1
<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>\ifnomaxlines</code> .....	1
<code>\ifnosyncpstarts</code> .....	1
<code>\ifPages@mainmatter</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>\ifsameparallelpagenumber</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>\ld@set</code> .....	1
<code>\ld@bfnote</code> .....	1
<code>\ldc@maxchunks</code> .....	1
<code>\ldcalc@maxoftwo</code> .....	1
<code>\ldcalc@minoftwo</code> .....	1
<code>\ldcalcnum</code> .....	1
<code>\ldchecklang</code> .....	1
<code>\ldleftbox</code> .....	1
<code>\ldlinenumR</code> .....	1
<code>\ldmake@labelsR</code> .....	1
<code>\ldminpagelines</code> .....	1
<code>\ldnumpstartsL</code> .....	1
<code>\ldnumpstartsR</code> .....	1
<code>\ldpagefullfalse</code> .....	1
<code>\ldpagefulltrue</code> .....	1
<code>\ldrightbox</code> .....	1
<code>\ldsamepagefalse</code> .....	1
<code>\ldsamepagetrue</code> .....	1
<code>\ldsetupmaxlinecounts</code> .....	1
<code>\ldsetuprawboxes</code> .....	1
<code>\ldskipversenumberR</code> .....	1
<code>\ldusedbabelfalse</code> .....	1
<code>\ldusedbabeltrue</code> .....	1
<code>\lduselanguage</code> .....	1
<code>\ldzeromaxlinecounts</code> .....	1
<code>\ldpscL</code> .....	1
<code>\ldpscR</code> .....	1
<code>\labelref@listR</code> .....	1
<code>\last@page@numR</code> .....	1
<code>\Lcolwidth</code> .....	1, 8, 10
<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@memnum</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@pbnunR</code>	1
<code>\led@pbR</code>	1
<code>\led@warn@ChangeSyncOption</code>	1
<code>\led@warn@setting@in@rightside</code>	1
<code>\lednopbnum</code>	1
<code>\lednopbnumR</code>	1
<code>\ledpbnunR</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>	14
<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, 15
<code>\lineationR</code>	1, 15
<code>\linenum@outR</code>	1
<code>\linenumberstyle*</code>	1, 15
<code>\linenumberstyleR</code>	1, 15
<code>\linenumincrement</code>	1, 15
<code>\linenumincrement*</code>	1, 15
<code>\linenumincrementR</code>	1, 15
<code>\linenummargin</code>	1
<code>\linenummargin*</code>	1, 16
<code>\linenummarginR</code>	1, 16
<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, 7
<code>\maxlinesinpar@list</code>	1
<code>\memorydump</code>	14
<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> .....	9
<code>\num@linesR</code> .....	1
<code>\numberpstartfalse</code> .....	15
<code>\numberpstarttrue</code> .....	15
<code>\numpagelinesL</code> .....	1
<code>\numpagelinesR</code> .....	1

## O

<code>\one@lineR</code> .....	1
<code>\onlysideX</code> .....	13
<code>optionadvancedshiftedpstarts</code> .....	10, 11
<code>optionnomaxlines</code> .....	10, 11, 21
<code>optionnosyncpstarts</code> .....	11, 21, 101
<code>optionshiftedpstarts</code> .....	6, 11, 21

## P

<code>\page@action</code> .....	1
<code>\page@numR</code> .....	1
<code>\Pages</code> .....	1, 9
<code>pages (environment)</code> .....	9
<code>\Pages@mainmatter</code> .....	1
<code>pairs (environment)</code> .....	8
<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> .....	16
<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> .....	16
<code>\pstartL</code> .....	1
<code>\pstartR</code> .....	1

## R

<code>\Rcolwidth</code> .....	1, 8, 10
<code>\read@linelist</code> .....	1
<code>\reledpar@error</code> .....	1
<code>\reledpar@warning</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
<code>Rightside (environment)</code> .....	14
<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, 12
<code>\sethangingsymbol</code> .....	17
<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> .....	16
<code>\setwidthliketwocolumns@C</code> .....	1

<code>\setwidthliketwocolumns@L</code> .....	<u>1</u>
<code>\setwidthliketwocolumns@R</code> .....	<u>1</u>
<code>\sidenote@marginR</code> .....	<u>1</u>
<code>\sidenotemargin*</code> .....	<u>1</u>
<code>\skip@lockoff</code> .....	<u>1</u>
<code>\skipnumbering</code> .....	<u>1</u> , 15
<code>\startlock</code> .....	<u>1</u>
<code>\startsub</code> .....	<u>1</u>
<code>\sub@action</code> .....	<u>1</u>
<code>\subline@numR</code> .....	<u>1</u>
<code>\sublinenumberstyle*</code> .....	<u>1</u> , 15
<code>\sublinenumberstyleR</code> .....	<u>1</u> , 15
<code>\sublinenumincrement</code> .....	<u>1</u> , 15
<code>\sublinenumincrement*</code> .....	<u>1</u> , 15
<code>\sublinenumincrementR</code> .....	<u>1</u> , 15
<code>\sublinenumrepR</code> .....	<u>1</u>

## T

<code>\theledlanguageL</code> .....	<u>1</u>
<code>\theledlanguageR</code> .....	<u>1</u>
<code>\thepar@page</code> .....	<u>1</u>
<code>\thepstartL</code> .....	15
<code>\thepstartR</code> .....	15
<code>\thestanzaL</code> .....	<u>1</u> , 18
<code>\thestanzaR</code> .....	<u>1</u> , 18

## U

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

## W

<code>\widthliketwocolumns</code> .....	9
---	---

## X

<code>\Xnoteswidthliketwocolumns</code> .....	9
<code>\Xonlyside</code> .....	13

## Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	75
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	81
\Pages: Added \l@duselanguage to \Pages	91
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	55
Added hooks into Leftside environment	46
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match neweledmac	59
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	58
\do@lineL: Added \do@lineLhook to \do@lineL	53
Simplified \do@lineL by using macros for some common code	53
\do@lineR: Changed \do@lineR similarly to \do@lineL	55
\flag@end: Removed extraneous spaces from \flag@end	41
\ifledRcol: Moved \ifl@dpairing toeledmac	22
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac	25
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	32
\l@dnumpstartsR: Moved \l@dnumpstartsL toeledmac	77
\ledstrutR: Added \ledstrutL and \ledstrutR	95
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX	72
\Pages: Added \ledstrutL to \Pages	91
Added \ledstrutR to \Pages	92
\printlinesR: Simplified \printlinesR by using \setprintlines	65
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	47
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	31
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Do not just set \line@marginR in \linenummargin	29
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	94
v0.3.c.	
General: Compatibilty with Polyglossia	1
v0.4.0.	
General: No more ledparpatch. All patches are now in the main file.	1
v0.5.0.	
General: Corrections about \section and other titles in numbered sections	1
v0.6.0.	
General: Be able to use \chapter in parallel pages.	1
v0.7.0.	
General: Option ‘shiftedverses’ which make there is no blank between two parallel verses with inequal 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> . . . . .	15
Possibility to number the pstart with the commands <code>\numberpstarttrue</code> . . . . .	1
<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code> . . . . .	22
v0.9.1.	
General: The numbering of the pstarts restarts on each <code>\beginnumbering</code> . . . . .	1
v0.9.2.	
General: Debug : with <code>\Columns</code> , the hanging indentation now runs on the left columns and the hanging symbol is shown only when <code>\stanza</code> is used. . . . .	1
v0.9.3.	
General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with <code>memoir</code> class. . . . .	1
v0.10.0.	
General: <code>\edlabel</code> commands on the right side are now correctly indicated. . . . .	1
<code>\edlabel</code> commands which start a paragraph are now put in the right place. . . . .	1
v0.11.0.	
General: Change <code>\do@lineL</code> and <code>\do@lineR</code> to allow line numbering by pstart (like in <code>eledmac 0.15</code> ). . . . .	53
Lineation can be by pstart (like in <code>eledmac 0.15</code> ). . . . .	28
New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . .	72
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code> ). . . . .	59
<code>\Columns</code> : Line numbering by pstart. . . . .	82
<code>\get@nextboxR</code> : Change <code>\get@nextboxL</code> and <code>\get@nextboxR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code> ). . . . .	101
Pstart number can be printed in side . . . . .	103
<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	72
v0.12.0.	
General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . . . .	72
v1.0.0.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code> . . . . .	1
Debug in lineation by pstart . . . . .	28
v1.0.1.	
General: Correction on <code>\numberonlyfirstinline</code> with lineation by pstart or by page. .	1
v1.1.0.	
General: <code>Shiftedverses</code> becomes <code>shiftedpstarts</code> . . . . .	1
<code>\pstartR</code> : Add <code>\labelpstarttrue</code> (from <code>eledmac</code> ). . . . .	48
v1.1.1.	
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1. . . . .	48
v1.1.2.	
<code>\affixside@noteR</code> : Remove spurious space between line number and line content . .	71
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.	26

v1.3.0.	
<code>\edtext</code> : Manage RTL language. . . . .	41
v1.3.1.	
<code>\l@dbfnote</code> : Compatibility of standard footnotes with <code>eledmac</code> when theses footnotes contain any commands. . . . .	72
v1.3.2.	
General: Debug with some classes. . . . .	1
v1.3.3.	
General: Debugging the left notes of the right column. . . . .	71
<code>\l@dbfnote</code> : Spurious space with footnote in right column. . . . .	72
v1.3.4.	
General: Allow use of commands in sidenotes, as introduced by <code>eledmac</code> 1.0. . . . .	71
v1.3.5.	
<code>\normalbfnoteX</code> : Allows one to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded. . . . .	72
v1.4.0.	
General: Added <code>\do@insidelineLhook</code> and <code>\do@insidelineRhook</code> . . . . .	55
v1.4.1.	
General: Enable the use of <code>stanzaindentsrepetition</code> within <code>astanza</code> environment. . . . .	73
<code>\normalbfnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text. . . . .	72
v1.4.3.	
General: Corrects a false hanging verse when a verse is exactly the length of a line. . . . .	1
<code>\inserthangingsymbolR</code> : Hanging verse is no longer automatically flush right. . . . .	72
<code>\pendL</code> : Spurious spaces in <code>\pendL</code> . . . . .	51
<code>\pendR</code> : Spurious spaces in <code>\pstartR</code> . . . . .	51
<code>\pstartR</code> : Spurious spaces in <code>\pstartL</code> and <code>\pstartR</code> . . . . .	48
v1.5.0.	
General: Add, as in <code>eledmac</code> , features to manage page breaks. . . . .	1
<code>\sublinenumincrement*</code> : Add starred version of <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\firstsublinenum</code> , <code>\sublinenumincrement</code> to change both Left and Rightside. . . . .	30
v1.6.0.	
General: Add tool and documentation for parallel ledgroups . . . . .	18
v1.7.0.	
General: Add, as in <code>eledmac</code> , features to make crossrefs with <code>pstart</code> numbers. . . . .	1
v1.8.0.	
General: <code>\beginnumbering</code> is defined only on <code>eledmac</code> , not on <code>eledpar</code> . . . . .	25
<code>\l@dlsnote</code> , <code>\l@drsnote</code> and <code>\l@dcsnote</code> defined only one time, in <code>eledmac</code> . . . . .	71
Add <code>\beforecolumnseparator</code> and <code>\aftercolumnseparator</code> . . . . .	9
Add <code>\columnspostion</code> . . . . .	9
Add, as in <code>eledmac</code> , new system of sectioning commands. . . . .	1
Add, as in <code>eledmac</code> , option to insert something after <code>\pends</code> / verses. . . . .	1
Add, as in <code>eledmac</code> , option to insert something between <code>\pstarts</code> / verse. . . . .	1
Change <code>\do@lineR</code> and <code>\do@lineR</code> to allow new sectioning commands. . . . .	53
Compatibility with <code>musixtex</code> . . . . .	1
Debug <code>eledmac</code> sectioning command after using <code>\resumenumbering</code> . . . . .	1
New sectioning commands, as in <code>eledmac</code> . . . . .	19
Suppress <code>\ifl@dsamelang</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. . . . .	75
<code>\Columns</code> : Modify <code>\Columns</code> to enable to add section's title. . . . .	80



Suppress \l@dchecklang from \Columns. . . . .	81
\l@dchecklang: Suppress \l@dchecklang which did not work and was not logical, because both columns could have the same language but not the main language of the document. . . . .	75
\Pages: Modify \Pages to enable to add section's title. . . . .	88
\pendL: As in eledmac, \pendL can have an optional argument. . . . .	51
\pendR: As in eledmac, \pendR can have an optional argument. . . . .	51
\print@columnseparator: Move some code of \Columns to \print@columnseparator. . . . .	83
\pstartR: As in eledmac, \pendL and \pendR can have an optional argument. . . . .	48
\sidenotemargin*: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar. . . . .	71
Add \sidenotemargin* . . . . .	71
\theledlanguageR: Correct left/right language setting with polyglossia. . . . .	77
v1.8.1.	
\do@lineL: Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?). . . . .	53
\do@lineR: Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?). . . . .	55
v1.8.2.	
General: Debug \eledxxx with some paper sizes . . . . .	1
Debug left and side note (bugs added by 1.8.0) . . . . .	1
\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar . . . . .	41
\lineation*: Add \lineation* . . . . .	29
\reledpar@error: Errors specific to eledpar send to eledpar handbook . . . . .	23
v1.8.3.	
General: Add \noeledxxx, as in eledmac . . . . .	1
\doinsidelineRhook: Added \dolineLhook, \dolineRhook, \doinsidelineLhook and \doinsidelineRhook . . . . .	55
\Pages: Debug blank pages when using optional argument in the last \pend. . . . .	88
\resumenumberingR: Debug \resumenumberingR . . . . .	27
v1.9.0.	
General: Add \AtBeginPairs macro. . . . .	8
Compatibility with \Xnoteswidthliketwocolumns and \notesXwidthliketwocolumns . . . . .	1
\ifwidthliketwocolumns: Added widthliketwocolumns option . . . . .	22
\theledlanguageR: Debug left/right language switching with polyglossia. Do not write in .aux file when setting left/right lines. . . . .	77
v1.9.1.	
\ifledRcol: Moved \ifl@dpaging to eledmac . . . . .	22
v1.10.0.	
General: Compatibility with \AtEveryPstart and \AtEveryPend . . . . .	1
Restore critical notes in \eledsection in parallel columns (this bug was added in 1.8.2). . . . .	1
\Pages: Debug wrong pages splitting when no optional argument is used in last \pend (bug was added in v1.8.3). . . . .	88
Debug wrong parallel pages synchronization when an \edtext falls across two pages. . . . .	88
v1.10.1.	
\line@list@stuffR: Revert modification of 1.4.2, which makes bugs with numbering. Leave vertical mode to solve spurious space before minipage. . . . .	40

v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles. . . . .	1
\edtext: \critext and \edtext are now defined only in eledmac. . . . .	41
v1.12.0.	
General: Compatibility with Lua $\TeX$ RTL languages. . . . .	1
\Columns: Add \l@dprintingcolumnstrue . . . . .	80
\edlabel: \edlabel and \edindex works now with hyperref when using eledpar. . .	70
\edlabel is now defined only one time for both eledmac and eledpar . . . . .	70
\Pages: Add \l@dprintingpagestrue . . . . .	88
\print@eledsectionL: Compatibility with Lua $\TeX$ RTL languages. . . . .	54
\print@eledsectionR: Compatibility with Lua $\TeX$ RTL languages. . . . .	56
\print@lineL: Compatibility with Lua $\TeX$ RTL languages. . . . .	54
v1.12.1.	
\print@eledsectionL: Fixes bug with Lua $\TeX$ RTL \eledsection. . . . .	54
v1.13.0.	
General: Enable the use of optional argument of & in astanza environment. . . . .	73
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@drihtpage: Use \newpage instead of \clearpage. . . . .	96
\ifledRcol: Remove false boolean settings which are not needed. . . . .	22
\Pages: Prevent false overfull hboxes when using \Pages outside of pages environment.	89
When using shiftedpstarts option, a \l@dleftbox with a null height will advance the \pagetotal in any case. . . . .	88
v1.13.1.	
\correct@footinsX@box: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages. . . . .	66
Correct \correct@footinsX@box and \correct@Xfootins@box . . . . .	66
\Pages: Prevent false empty page after \Pages (bug added in 1.13.0) . . . . .	88
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. . . . .	12
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 . . . . .	58
\inserthangingsymbolR: Prevent more efficiently the column separator from shifting when a verse is hanging . . . . .	72
\lineationR: As \lineation, \lineationR automatically set the \pstartinfootnote. . . . . .	28
\n@num: \n@num defined only one time for both Eledmac and Eledpar. . . . .	37
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar . . . . .	41

v1.16.0.	
General: Error message when calling \Pages inside 'pages' environment and \Columns 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 \beginnumbering...\endnumbering without \pstart. . .	1
Fix bug with nofamiliar / nocritical option of eledmac. ....	1
New package option sameparallelpagenummer to have the same page number for both left and right side. ....	1
\newseries@par: Fix bug with \onlysideX. ....	42
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 \pend in combination with some specific penalties setting. ....	1
v1.17.0.	
General: Add compatibility of optional argument of \pstart/\pend and \AtEveryPstart/\AtEveryPend with two columns mode. ....	1
v1.21.0.	
General: Add \hidenummering ....	15
v2.0.0.	
\@adv: \@adv defined only in reledmac. ....	35
\@lab: \@lab defined only in eledmac. ....	70
\@ref@regR: \@ref defined only in reledmac, code specific to right side moved in \ref@regR. ....	37
\@set: \@set defined only in reledmac. ....	35
General: \@nl is now defined only in reledmac. ....	34
\ifbypage@ and \ifbypstart@R defined in eledmac. ....	28
Fix some bugs with 'sameparallelpagenummer' option. ....	1
Many code refactored and moved to reledmac. ....	1
Package's name becomes reledpar. ....	1
Totally new implementation of 'sameparallelpagenummer' option. ....	1
\advanceline: \advanceline defined only in reledmac. ....	41
\bbl@set@language: Patch \bbl@set@language instead of redefining it ....	75
\do@lockonR: \do@lockon defined only in reledmac. ....	36
\endlock: \startlock and \endlock defined only in reledmac. ....	41
\endsub: \startsub and \endsub defined only in reledmac. ....	41
\fix@page: \fix@page is defined only once in reledmac ....	35
chapterinpages: Deleting the old system of managing parallel chapter, keep only the new one with \patchcmd. ....	46
\l@d@set: \l@d@set defined only in reledmac. ....	35
\l@dbfnote: \l@dbfnote defined only in reledmac. ....	72
\line@marginR: \linenummargin now defined only once time in reledmac. ....	29
\normalbfnoteX: \normalbfnoteX defined only in reledmac. ....	72
\page@action: \page@action defined only in reledmac. ....	35
\read@linelist: \read@linelist is defined only once time in \reledmac. ....	34
\set@line: \set@line defined only in reledmac. ....	41
\set@line@action: \set@line@action defined only in reledmac. ....	35
\setline: \setline defined only in reledmac. ....	41
\setlinenum: \setlinenum defined only in reledmac. ....	41

<code>\skip@lockoff: \do@lockoff</code> defined only in <code>reledmac</code> . . . . .	36
<code>\sub@action: \sub@action</code> defined only in <code>reledmac</code> . . . . .	35
<code>\sublinenumincrement*: \firstlinenum, \linenumincrement, \firstsublinenum,</code> <code>\sublinenumincrement</code> are now defined only in <code>reledmac</code> . . . . .	30
<code>\theledlanguageR: Patch \otherlanguage</code> instead of redefining it. . . . .	77
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>\setparledgroupnotespacing</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 $\TeX$ release of october 2015 . . . . .	1
Fix bug with title number in parallel columns . . . . .	1
New line setting command suffixed by <code>R</code> to set only the right side. . . . .	1
<code>\Pages: Fix bug</code> when calling <code>\Columns</code> after a <code>\Pages</code> (bug added in v1.13.0). . . . .	89
v2.4.0.	
General: New way of (not) synchronizing the parallel pages. . . . .	1
Option to switch to <code>\mainmatter</code> when calling <code>\Pages</code> . . . . .	1
<code>\ledstrutR: Deleted \ledtrutL and \ledstrutR</code> . . . . .	95
Fix bug with dotted letter . . . . .	95
v2.5.0.	
General: Disable empty lines as paragraph in <code>astanza</code> . . . . .	1
Fix bug introduced in v1.15.0 which made hanging indentation in verse not work any- more. . . . .	1
New commands <code>\linenummarginR</code> and <code>\linenummargin*</code> . . . . .	1
v2.5.1.	
General: Fix spurious space when using optional argument of <code>astanza</code> environment (in- troduced in v2.5.0). . . . .	1