

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

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

Abstract

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

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

To report bugs, please go to `ledmac`’s GitHub page and click “New Issue”: <https://github.com/maieul/ledmac/issues/>. You must open an account with github.com to access my page ([maieul/ledmac](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

1 Introduction	5
1.1 Aim of this package	5
1.2 Historical overview	5
2 Options	6
2.1 Synchronization’s options	6
2.2 Other options	6
3 General	6

*This file (`reledpar.dtx`) has version number v2.4.0, last revised 2015/09/29.

[†]`maieul at maieul dot net`

[‡]`herries dot press at earthlink dot net`

4 Parallel columns	8
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
5 Facing pages	9
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
6 Left and right texts	14
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	16
7 Verse	17
8 Side notes	18
9 Parallel ledgroups	18
9.1 General	18
9.2 Parallel ledgroups and setspace package	19
10 Sectioning commands	19
11 Notes about page number	19
I Implementation overview	20

II Preliminaries	20
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
III Sectioning commands	24
IV Line counting	28
IV.1 Setting lineation reset	28
IV.2 Setting line number margin	29
IV.3 Setting lineation start and step	29
IV.4 Setting line flag	30
IV.5 Setting line number style	31
IV.6 Print marginal line number	31
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>	32
IV.8 Reading the line-list file	33
IV.9 Commands within the line-list file	33
IV.10 Writing to the line-list file	39
V Marking text for notes	41
V.1 Specific hooks and commands for notes	41
V.1.1 Notes to be printed on one side only	41
V.1.2 Familiar footnotes without marks	42
V.1.3 Create hooks	43
V.1.4 Init standards series (A,B,C,D,E,Z)	44
VI Pstart numbers dumping and restoration	44
VII Parallel environments	45
VIII Paragraph decomposition and reassembly	47
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	47
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	63
VIII.8 Printing leftover notes	64

IX Footnotes	65
IX.1 Line number printing	65
IX.2 Footnotes output specific to \Pages	65
X Cross referencing	70
XI Side notes	70
XII Familiar footnotes	72
XIII Verse	72
XIV Naming macros	74
XV Fixing babel and polyglossia	75
XVI Counts and boxes for parallel texts	77
XVII Checking text to be processed	78
XVIII Parallel columns	80
XIX Parallel pages	88
XIX.1 Specific counters	88
XIX.2 Main macro	88
XIX.3 Ensure all notes be printed at the end of parallel pages	94
XIX.4 Struts	95
XIX.5 Page clearing	95
XIX.6 Lines managing	96
XIX.7 Page break managing	98
XIX.8 Getting boxes content	101
XX Page numbering	105
XX.1 Global options	105
XX.2 mainmatter option of \Pages	107
XXI Sections' titles' commands	107
XXII Page break/no page break, depending on the specific line	108
XXIII Parallel ledgroup	109
XXIV Compatibility with eledmac	113
XXV The End	113

Appendix A Some things to do when changing version	114
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>\renewcommand</code> replaced 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
References	115
Index	115
Change History	133

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`.¹

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.²

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

¹This option can either be used on `reledmac` or `reledpar`.

²This 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 parallel³. 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.

³You 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:

- 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 synchronizing⁴

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:

⁴There 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 `sameparallelpagelnumber` 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.⁵ 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 ...
```

⁵The 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_EX'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}

```

It is possible to insert a number at every `\pstart` command. You must use the `\numberpstarttrue` command to have it. You can stop the numbering with `\numberpstartfalse`. 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. Using these macros inside the `Leftside` and `Rightside` environments gives you independent control over the left and right numbering schemes. The `\firstsublinenum` and `\sublinenumincrement` macros correspondingly set the numbering scheme for sublines. Generally speaking, controls like `\firstlinenum` or `\linenummargin` apply to sequential and left texts. To effect right texts only, they have to be within a `Rightside` environment.

```

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

```

The starred versions change both left and right numbering schemes. The suffixed version change the right side, without regard to the position they are called.

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

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

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

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

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. 15).

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 impor-

tant to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default. Also, it is the *last* language setting in a side that controls the language used in any notes for that side when they get printed. If you are using multilingual notes then it is probably safest to explicitly specify the language(s) for each note rather than relying on the language selection for the side. The right side language is also applied to the right side line numbers.

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 letter syncs verse by verse, while the environment syncs stanza by stanza.

If you get an error message along the lines of 'Missing number, treated as zero `\sza@0@`' it is because you have forgotten to use `\setstanzaindent` to set the stanza indents.

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

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

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

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

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

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

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

`\thestanzaL` When using `\stanzanumtrue` (8.9 p. 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

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

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

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

II Preliminaries

II.1 Package’s meta-data

Announce the name and version of the package, which is targeted for $\text{\LaTeX}2\text{\epsilon}$. 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}[2015/09/29 v2.4.0 reledmac extension for
  parallel texts]%
4
5 %
```

II.2 Package’s requirement

Few commands use `\xspace` command.

```

6 \RequirePackage{xspace}%
7 %
```

II.3 Package’s options

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

```

8 \RequirePackage{xkeyval}
9 %
```

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   \nomaxlinestrue%
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 %
```

```
d@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 %
```

```
ftside@PreviousNotPrinted74 \newcommand*\led@err@Leftside@PreviousNotPrinted{%
htside@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 }%

```

```

\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 %

```

III Sectioning commands

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

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


```

104 \newcount\section@numR
105 \section@numR=\z@
106 %

```

`\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`.

```

107 \pst@rtedLfalse
108 \newif\ifpst@rtedR
109
110 %

```

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

```

111 \newcommand*{\beginnumberingR}{%
112   \ifnumberingR
113     \led@err@NumberingStarted
114     \endnumberingR
115   \fi
116   \global\l@dnumpstartsR \z@
117   \global\pst@rtedRfalse
118   \global\numberingRtrue
119   \global\advance\section@numR \@ne
120   \global\absline@numR \z@
121   \gdef\normal@page@breakR{}
122   \gdef\l@prev@pbR{}
123   \gdef\l@prev@nopbR{}
124   \global\line@numR \z@
125   \global\@lockR \z@
126   \global\sub@lockR \z@
127   \global\sublines@false
128   \global\let\next@page@numR\relax
129   \global\let\sub@change\relax
130   \message{Section \the\section@numR R }%
131   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
132   \l@dend@stuff
133   \setcounter{pstartR}{1}
134   \begingroup
135   \initnumbering@sectcountR
136   \gdef\eled@sectionsR@{ }%
137   \if@noeled@sec\else%
138     \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@numR R
139     }{}{} \makeatother%
140     \immediate\openout\eled@sectioningR@out=\jobname.eledsec\the\
141     section@numR R\relax%
142   \fi%
143 }
144 %

```

`\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.

```

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

```

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

```

173 \newcounter{chapterR}
174 \newcounter{sectionR}
175 \newcounter{subsectionR}
176 \newcounter{subsubsectionR}
177 \newcommand{\initnumbering@sectcountR}{
178   \let\c@chapter\c@chapterR
179   \let\c@section\c@sectionR
180   \let\c@subsection\c@subsectionR
181   \let\c@subsubsection\c@subsubsectionR

```

```

182 }
183 %

```

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

```

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

```

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

```

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

```

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`.

```

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

```

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

```

255 \WithSuffix\newcommand\lineation*[1]{%

```

```

256 \lineation{#1}%
257 \lineationR{#1}%
258 }%
259 %

```

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.

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`.

```

260 \newcount\line@marginR
261 %

```

By default put right text numbers at the right.

```

262 \line@marginR=\@ne
263
264 %

```

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.

```

265 \newcounter{firstlinenumR}
266 \setcounter{firstlinenumR}{5}
267 \newcounter{linenumincrementR}
268 \setcounter{linenumincrementR}{5}
269 %

```

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

```

270 \newcounter{firstsublinenumR}
271 \setcounter{firstsublinenumR}{5}
272 \newcounter{sublinenumincrementR}
273 \setcounter{sublinenumincrementR}{5}
274
275 %

```

`\firstlinenum` These are the user's macros for changing (sub) line numbers. They are defined in
`\linenumincrement` reledmac. The starred versions are specific to eledpar.
`\firstsublinenum`
`\sublinenumincrement` 276 `\WithSuffix\newcommand\firstlinenum*[1]{%`
`\firstlinenum*` 277 `\setcounter{firstlinenumR}{#1}%`
`\linenumincrement*` 278 `\setcounter{firstlinenum}{#1}%`
`\firstsublinenum*` 279 `}`
`\sublinenumincrement*` 280 `\WithSuffix\newcommand\linenumincrement*[1]{%`
`\sublinenumincrement*` 281 `\setcounter{linenumincrementR}{#1}%`
282 `\setcounter{linenumincrement}{#1}%`
283 `}`
284 `\WithSuffix\newcommand\firstsublinenum*[1]{%`
285 `\setcounter{subfirstlinenumR}{#1}%`
286 `\setcounter{subfirstlinenum}{#1}%`
287 `}`
288 `\WithSuffix\newcommand\sublinenumincrement*[1]{%`
289 `\setcounter{sublinenumincrementR}{#1}%`
290 `\setcounter{sublinenumincrement}{#1}%`
291 `}`
292 `%`

`\firstlinenumR` And the 'R' suffixed version.
`\linenumincrementR`
`\firstsublinenumR` 293 `\newcommand\firstlinenumR[1]{%`
`\sublinenumincrementR` 294 `\setcounter{firstlinenumR}{#1}%`
295 `}`
296 `\newcommand\linenumincrementR[1]{%`
297 `\setcounter{linenumincrementR}{#1}%`
298 `}`
299 `\newcommand\firstsublinenumR[1]{%`
300 `\setcounter{subfirstlinenumR}{#1}%`
301 `}`
302 `\newcommand\sublinenumincrementR[1]{%`
303 `\setcounter{sublinenumincrementR}{#1}%`
304 `}`
305 `%`

IV.4 Setting line flag

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

```

306 \newcommand{\setRlineflag}[1]{%
307   \gdef\@Rlineflag{#1}%
308 }
309 \setRlineflag{R}
310 %

```

IV.5 Setting line number style

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

```

311 \newcommand*{\linenumrepR}[1]{\@arabic{#1}}
312 \newcommand*{\sublinenumrepR}[1]{\@arabic{#1}}
313
314 %

```

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

```

315 \newcommand*{\linenumberstyleR}[1]{%
316   \def\linenumrepR##1{\@nameuse{#1}{##1}}
317 \newcommand*{\sublinenumberstyleR}[1]{%
318   \def\sublinenumrepR##1{\@nameuse{#1}{##1}}
319 %

```

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

```

320 \WithSuffix\newcommand\linenumberstyle*[1]{%
321   \linenumberstyle{#1}%
322   \linenumberstyleR{#1}%
323 }%
324
325 \WithSuffix\newcommand\sublinenumberstyle*[1]{%
326   \sublinenumberstyle{#1}%
327   \sublinenumberstyleR{#1}%
328 }%
329 %
330 %

```

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`.

```

331 \newcommand*{\leftlinenumR}{%
332   \l@dlinenumR
333   \kern\linenumsep}
334 \newcommand*{\rightlinenumR}{%
335   \kern\linenumsep
336   \l@dlinenumR}
337 \newcommand*{\l@dlinenumR}{%
338   \numlabfont\linenumrepR{\line@numR}\@Rlineflag%
339   \ifsublines@
340     \ifnum\subline@num>\z@
341       \unskip\fullstop\sublinenumrepR{\subline@num}%
342     \fi

```

```

343 \fi}
344
345 %

```

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.

```

346 \newcount\line@numR
347 \newcount\subline@numR
348 \newcount\absline@numR
349
350 %

```

`\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`

```

351 \list@create{\line@listR}
352 \list@create{\insertlines@listR}
353 \list@create{\actionlines@listR}
354 \list@create{\actions@listR}
355
356 %

```

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

```

357 \newcount\page@numR
358
359 %

```

IV.7.2 Specific to `reledpar`

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

`\linesinpar@listR`

`\maxlinesinpar@list`


```

360 \list@create{\linesinpar@listL}
361 \list@create{\linesinpar@listR}
362 \list@create{\maxlinesinpar@list}
363
364 %

```

IV.8 Reading the line-list file

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

```

365 \newcommand{\list@clearing@regR}{%
366   \list@clear{\line@listR}%
367   \list@clear{\insertlines@listR}%
368   \list@clear{\actionlines@listR}%
369   \list@clear{\actions@listR}%
370   \list@clear{\linesinpar@listR}%
371   \list@clear{\linesonpage@listR}
372 }
373 %

```

\@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.

```

374 \newcommand{\@par@sync@option}[1]{%
375   \IfStrEq{#1}{\@par@this@sync@option}%
376   {}%
377   {\ifledRcol%
378     \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
379   }%
380   \else%
381     \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
382   }%
383   \fi%
384   \endinput%
385 }%

```

\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.

```

386 \newcommand{\@nl@regR}{%
387   \ifx\l@dchset@num\relax \else
388     \advance\absline@numR \@ne
389     \set@line@action
390     \let\l@dchset@num\relax
391     \advance\absline@numR \m@ne
392     \advance\line@numR \m@ne%    % do we need this?
393   \fi
394   \advance\absline@numR \@ne
395   \ifx\next@page@numR\relax \else
396     \page@action
397     \let\next@page@numR\relax
398   \fi
399   \ifx\sub@change\relax \else
400     \ifnum\sub@change>\z@
401       \sublines@true
402     \else
403       \sublines@false
404     \fi
405     \sub@action
406     \let\sub@change\relax
407   \fi
408   \ifcase\@lockR
409   \or
410     \@lockR \tw@
411   \or\or
412     \@lockR \z@
413   \fi
414   \ifcase\sub@lockR
415   \or
416     \sub@lockR \tw@
417   \or\or
418     \sub@lockR \z@
419   \fi
420   \ifsublines@
421     \ifnum\sub@lockR<\tw@
422       \advance\subline@numR \@ne
423     \fi
424   \else
425     \ifnum\@lockR<\tw@
426       \advance\line@numR \@ne \subline@numR \z@
427     \fi
428   \fi}
429
430
431 %

```

`\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`.

```

432 \newcount\last@page@numR
433 \last@page@numR=-10000
434
435 %

```

`\@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`.

```

436 \newcount\@lockR
437 \newcount\sub@lockR
438
439 \newcommand*{\do@lockonR}{%
440   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
441   \ifsublines@
442     \xright@appenditem{-1005}\to\actions@listR
443     \ifnum\sub@lockR=\z@
444       \sub@lockR \@ne
445     \else
446       \ifnum\sub@lockR=\thr@@
447         \sub@lockR \@ne
448       \fi
449     \fi
450   \else
451     \xright@appenditem{-1003}\to\actions@listR
452     \ifnum\@lockR=\z@

```

```

453 \@lockR \@ne
454 \else
455 \ifnum\@lockR=\thr@@
456 \@lockR \@ne
457 \fi
458 \fi
459 \fi}
460
461 %

```

`\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@lockoff` `\do@lockoffR` `\do@lockoffR`.
`\skip@lockoff`

```

462
463
464 \newcommand{\do@lockoffR}{%
465 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
466 \ifsublines@
467 \xright@appenditem{-1006}\to\actions@listR
468 \ifnum\sub@lockR=\tw@
469 \sub@lockR \thr@@
470 \else
471 \sub@lockR \z@
472 \fi
473 \else
474 \xright@appenditem{-1004}\to\actions@listR
475 \ifnum\@lockR=\tw@
476 \@lockR \thr@@
477 \else
478 \@lockR \z@
479 \fi
480 \fi}
481
482
483 %

```

`\n@num`

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

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

```

484 \newcount\insert@countR
485 %

```

- #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.

```

486 \newcommand*{\@ref@regR}[2]{%
487   \global\advance\@edtext@level by 1%
488   \global\insert@countR=#1\relax
489   \loop\ifnum\insert@countR>\z@
490     \xright@appenditem{\the\absline@numR}\to\insertlines@listR
491     \global\advance\insert@countR \m@ne
492   \repeat
493 %

```

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.

```

494 \begingroup
495   \let\@ref=\dummy@ref
496   \let\@lopR\@gobble
497   \let\page@action=\relax
498   \let\sub@action=\relax
499   \let\set@line@action=\relax
500   \let\@lab=\relax
501   \let\@lemma=\relax
502   \let\@sw\@gobblethree%
503   #2
504   \global\endpage@num=\page@numR
505   \global\endline@num=\line@numR
506   \global\endsubline@num=\subline@numR
507 \endgroup
508 %

```

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

```

509 \xright@appenditem%
510   {\the\page@numR|\the\line@numR|}%
511   \ifsublines@ \the\subline@numR \else 0\fi|}%
512   \the\endpage@num|\the\endline@num|}%
513   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
514 %

```

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

```

515 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@the\
@edtext@level\endcsname}%
516 %

Declare and init boolean for lemma in this level.

517 \providebool{lemmacommand@the\@edtext@level}%
518 \boolfalse{lemmacommand@the\@edtext@level}%
519 %

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

520 #2
521 % Now, we store the list of \protect\cs{sw} of this current \protect\cs{
edtext} as an element of
522 % the global list of list of \protect\cs{sw} for a \protect\cs{edtext}
depth.
523 \begin{macrocode}
524 \ifnum\@edtext@level>0%
525 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtextR@the\@edtext@level\endcsname}}%
526 \ifcsundef{sw@list@edtextR@the\@edtext@level}{\
create@this@edtext@level}{}%
527 \letcs{\@tmp}{sw@list@edtextR@the\@edtext@level}%
528 \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}%
529 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
530 \global\cslet{sw@list@edtextR@the\@edtext@level}{\@tmp}%
531 \fi%
532 %

Decrease edtext level counter.

533 \global\advance\@edtext@level by -1%
534 }
535 %

```

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

```

536 \renewcommand*{\@pend}[1]{%
537 \ifbypstart@global\line@num=0\fi%
538 \xright@appenditem{#1}\to\linesinpar@listL}
539 \renewcommand*{\@pendR}[1]{%
540 \ifbypstartR@global\line@numR=0\fi
541 \xright@appenditem{#1}\to\linesinpar@listR}
542
543 %

```

\@pstart \@pstart and cs@pstartR allows us to know, when using \nomaxlines option in which
\@pstartR 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

```

544 \newcommand{\@pstart}[3]{%
545   \ifcsdef{minpage@pstart@#1}%
546     {\ifnumgreater{#2}{\csuse{minpage@pstart@#1}}%
547       {\csnumgdef{minpage@pstart@#1}{#2}}%
548     }%
549   }%
550   {\csnumgdef{minpage@pstart@#1}{#2}}
551   \csnumgdef{afterlines@pstart@#1L}{#3}%
552 }%
553
554 \newcommand{\@pstartR}[3]{%
555   \numdef{\@tmp}{#2-1}%Because we have not to know in which page the pstart
556   starts, but in which pair of facing page
557   \ifcsdef{minpage@pstart@#1}%
558     {\ifnumgreater{\@tmp}{\csuse{minpage@pstart@#1}}%
559       {\csnumgdef{minpage@pstart@#1}{\@tmp}}%
560     }%
561     {\csnumgdef{minpage@pstart@#1}{\@tmp}}
562   \csnumgdef{afterlines@pstart@#1R}{#3}%
563 }%
564 %

```

`\@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`.

```

565 \renewcommand*{\@lopL}[1]{%
566   \xright@appenditem{#1}\to\linesonpage@listL}
567 \renewcommand*{\@lopR}[1]{%
568   \xright@appenditem{#1}\to\linesonpage@listR}
569
570 %

```

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`.

```

571 \newwrite\linenum@outR
572 %

```

`\iffirst@linenum@outR` 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@outRtrue`

`\first@linenum@outRfalse`

```

573 \newif\iffirst@linenum@out@R
574 \first@linenum@out@Rtrue
575 %

```

\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.

```

576 \newcommand*{\line@list@stuffR}[1]{%
577   \read@linelist{#1}%
578   \iffirst@linenum@out@R
579     \immediate\closeout\linenum@outR
580     \global\first@linenum@out@Rfalse
581     \immediate\openout\linenum@outR=#1
582     \immediate\write\linenum@outR{\string\line@list@version{\
this@line@list@version}}%
583     \ifl@dpaging%
584       \immediate\write\linenum@outR{\string\@par@sync@option{\
@par@this@sync@option}}%
585     \fi%
586   \else
587     \if@minipage%
588       \leavevmode%
589     \fi%
590     \closeout\linenum@outR%
591     \openout\linenum@outR=#1%
592   \fi}
593
594 %

```

\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.

```

595 \newcommand*{\new@lineL}{%
596   \write\linenum@out{\string\@nl[\the\c@page][\thepage]}}
597 %

```

\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.

```

598 \newcommand*{\new@lineR}{%
599   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}
600 %

```

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

V Marking text for notes

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

`\critext`
`\edtext`
`\set@line`

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

V.1 Specific hooks and commands for notes

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

```
\newseries@par01 \newcommand{\newseries@par}[1]{%
602 %
```

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.

```

603 \unless\ifnofamiliar@%
604 \csgdef{onlysideX@#1}{}%
605 \global\newbool{keepforsideX@#1}%
606 \fi%
607 \unless\ifnocritical@%
608 \global\newbool{keepforXside@#1}%
609 \csgdef{Xonlyside@#1}{}%
610 \fi%
611 %

```

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`.

```

612 \unless\ifnofamiliar@%
613 % So declare the list.
614 % \begin{macrocode}
615 \expandafter\list@create\csname footnote#1@mk\endcsname%
616 %

```

Then, declare the `\footnoteXnomk` command.

```

617 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
618 %

```

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

```

619 \toggletrue{nomk@}%
620 \csuse{footnote#1}{##1}%
621 \togglefalse{nomk@}%
622 %

```

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`.

```

623 \letcs{\@tmp}{footnote#1@mk}%
624 \numdef{\@tmpa}{\csuse{c@footnote#1}}%
625 \global\xright@appenditem{\@tmpa}{to\@tmp%
626 \global\cslet{footnote#1@mk}{\@tmp}%
627 }%
628 %

```

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

```

629 \expandafter\newcommand\csname footnote#1mk\endcsname{%
630 %

```

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

```

631 \letcs{\@tmp}{footnote#1@mk}%
632 \gl@p\@tmp\to\@tmpa%
633 \global\cslet{footnote#1@mk}{\@tmp}%
634 %

```

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

```

635 \letcs{\old@footnote}{c@footnote#1}%
636 \setcounter{footnote#1}{\@tmpa}%
637 %

```

Define the footnote mark and print it

```

638 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
639 \csuse{@footnotemark#1}%
640 %

```

Restore previous footnote counter and finally add space.

```

641 \setcounter{footnote#1}{\old@footnote}%
642 \xspace%
643 }%
644 %

```

End of tools for familiar notes without marks

```

645 \fi
646 %

```

End of \newseries@par.

```

647 }%
648 %

```

V.1.3 Create hooks

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

```

649 \unless\ifnocritical@%
650 \newhookcommand@series{Xonlyside}%
651 \fi%
652 \unless\ifnofamiliar@%
653 \newhookcommand@series{onlysideX}%
654 \fi
655
656
657 %

```

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`.

```

658 \newcommand{\init@series@par}{%
659   \def\do##1{\newseries@par{##1}}%
660   \dolistloop{\@series}%
661 }%
662 \init@series@par%
663 %

```

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@pc64 \list@create{\list@pstartL@pc}%
\list@pstartR@pc65 \list@create{\list@pstartR@pc}%
666 %

```

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@pc67 \def\dump@pstartL@pc{%
\dump@pstartR@pc68   \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
669   \global\cslet{numberpstart@L\the\l@dnumpstartsL}{\ifnumberpstart}%
670 }%
671
672 \def\dump@pstartR@pc{%
673   \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
674   \global\cslet{numberpstart@R\the\l@dnumpstartsR}{\ifnumberpstart}%
675 }%
676
677 %

```

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

```

\restore@pstartR@pc
678 \def\restore@pstartL@pc{%
679   \ifx\list@pstartL@pc\empty\else%

```

```

680 \gl@p\list@pstartL@pc\to\@temp%
681 \global\c@pstartL=\@temp%
682 \fi%
683 }%
684 \def\restore@pstartR@pc{%
685 \ifx\list@pstartR@pc\empty\else%
686 \gl@p\list@pstartR@pc\to\@temp%
687 \global\c@pstartR=\@temp%
688 \fi%
689 }%
690 %

```

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.

```

691 \newenvironment{pairs}{%}
692 \l@dpairingtrue
693 \l@dpagingfalse
694 \initnumbering@quote
695 \at@begin@pairs%
696 }{%
697 \l@dpairingfalse
698 }
699 %
700 %

```

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

```

701 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
702 \def\at@begin@pairs{}%
703
704 %

```

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.

```

705 \newenvironment{pages}{%
706 \l@dpairingtrue
707 \l@dpagingtrue
708 \initnumbering@quote
709 \setlength{\Lcolwidth}{\textwidth}%
710 \setlength{\Rcolwidth}{\textwidth}%

```

```

711 }{%
712   \l@dpairingfalse
713   \l@dpagingfalse
714 }
715
716 %

```

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

```

717 \newif\ifinstanzaL
718 \newif\ifinstanzaR
719 %

```

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.

```

720 \newenvironment{Leftside}{%
721   \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
722   \led@err@Leftside@PreviousNotPrinted%
723   \fi%
724   \ledRcolfalse
725   \setcounter{pstartL}{1}
726   \let\pstart\pstartL
727   \let\thepstart\thepstartL
728   \let\pend\pendL
729   \let\memorydump\memorydumpL
730   \Leftsidehook
731   \let\old@startstanza\@startstanza
732   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
733 }{
734   \Leftsidehookend}
735 %

```

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

```

736 \Leftsidehookend
737 \Rightsidehook
738 \Rightsidehookend
739 \newcommand*{\Leftsidehook}{}
740 \newcommand*{\Leftsidehookend}{}
741 \newcommand*{\Rightsidehook}{}
742 \newcommand*{\Rightsidehookend}{}
743 %

```

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.

```

742 \newenvironment{Rightside}{%
743   \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
744   \led@err@Rightside@PreviousNotPrinted%
745   \fi%
746   \ledRcoltrue
747   \let\beginnumbering\beginnumberingR
748   \let\endnumbering\endnumberingR
749   \let\pausenumbering\pausenumberingR
750   \let\resumenumbering\resumenumberingR
751   \let\memorydump\memorydumpR
752   \let\thepstart\thepstartR
753   \let\pstart\pstartR
754   \let\pend\pendR
755   \let\ledpb\ledpbR
756   \let\lednopb\lednopbR
757   \let\lineation\lineationR
758   \Rightsidehook
759   \let\old@startstanza\@startstanza
760   \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
761 }{%
762   \ledRcolfalse
763   \Rightsidehookend
764 }
765
766 %

```

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.

```

767 \newcount\num@linesR

```

```

768 \newbox\one@lineR
769 \newcount\par@lineR
770 %

```

\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.

```

771
772 \newcounter{pstartL}
773 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
774 \newcounter{pstartR}
775 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
776
777 \newcommandx*{\pstartL}[1][1]{%
778   \if@nobreak%
779     \let\@oldnobreak\@nobreaktrue%
780   \else%
781     \let\@oldnobreak\@nobreakfalse%
782   \fi%
783   \@nobreaktrue%
784   \ifluatex%
785     \xdef\l@luatextextdir@L{\the\textdir}%
786     \xdef\l@luatexpardir@L{\the\pardir}%
787     \xdef\l@luatexbodydir@L{\the\bodydir}%
788   \fi%
789   \ifnumbering \else%
790     \led@err@PstartNotNumbered%
791     \beginnumbering%
792   \fi%
793   \ifnumberedpar@%
794     \led@err@PstartInPstart%
795     \pend%
796   \fi%
797 %

```

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

```

798 \ifpst@rtedL\else%
799   \list@clear{\inserts@list}%
800   \global\let\next@insert=\empty%
801   \global\pst@rtedLtrue%
802 \fi%
803 \begingroup\normal@pars%

```


804 %

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.

```
805 \global\advance\l@dnumpstartsL \@ne%
806 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
807   \led@err@TooManyPstarts%
808   \global\l@dnumpstartsL=\l@dc@maxchunks%
809 \fi%
810 \global\setnamebox{\l@dc@l@rawbox\the\l@dnumpstartsL}=\vbox\bgroup%
811 %
```

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.

```
812 \l@dzero@penalties%
813 \ifautopar\else%
814   \ifnumberpstart%
815     \ifsidepstartnum%
816       \else%
817         \thepstartL%
818       \fi%
819     \fi%
820   \fi%
821   \hsize=\Lcolwidth%
822   \numberedpar@true%
823   \iflabelpstart\protected@edef\@currentlabel%
824     {\p@pstartL\thepstartL}\fi%
825 %
```

Dump the optional arguments

```
826 \ifstrempy{#1}%
827   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\at@every@pstart}}%
828   {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent#1}}%
829   \at@every@pstart@call%
830 %
```

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

```
831 \ignorespaces%
832 %
```

```
833 }
834 %
```

The same for right side.

```
835 \newcommandx*{\pstartR}[1][1]{%
836   \if@nobraek%
837     \let\@oldnobraek\@nobraektrue%
838   \else%
```

```

839 \let\@oldnobreak\@nobreakfalse%
840 \fi%
841 \@nobreaktrue%
842 \ifluatex%
843 \xdef\l@luatextextdir@R{\the\textdir}%
844 \xdef\l@luatexpardir@R{\the\pardir}%
845 \xdef\l@luatexbodydir@R{\the\bodydir}%
846 \fi%
847 \ifnumberingR \else%
848 \led@err@PstartNotNumbered%
849 \beginnumberingR%
850 \fi%
851 \ifnumberedpar@%
852 \led@err@PstartInPstart%
853 \pendR%
854 \fi%
855 \ifpst@rtedR\else%
856 \list@clear{\inserts@listR}%
857 \global\let\next@insertR=\empty%
858 \global\pst@rtedRtrue%
859 \fi%
860 \begingroup\normal@pars%
861 \global\advance\l@dnumpstartsR \@one%
862 \ifnum\l@dnumpstartsR>\l@dc@maxchunks%
863 \led@err@TooManyPstarts%
864 \global\l@dnumpstartsR=\l@dc@maxchunks%
865 \fi%
866 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
867 \l@dzeropenalties%
868 \ifautopar\else%
869 \ifnumberpstart%
870 \ifsidepstartnum\else%
871 \thepstartR%
872 \fi%
873 \fi%
874 \fi%
875 \hsize=\Rcolwidth%
876 \numberedpar@true%
877 \iflabelpstart\protected@edef\@currentlabel%
878 {\p@pstartR\thepstartR}\fi%
879 \ifstrempy{#1}%
880 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
881 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
882 \at@every@pstart@call%
883 \ignorespaces%
884 }
885 %

```

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

```

886 \newcommandx*{\pendL}[1][1]{%
887   \ifnumbering \else%
888     \led@err@PendNotNumbered%
889   \fi%
890   \ifnumberedpar@ \else%
891     \led@err@PendNoPstart%
892   \fi%
893 %

```

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.

```

894 \endgraf\global\num@lines=\prevgraf\egroup%
895 \global\par@line=0%
896 %

```

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

```

897 \endgroup%
898 \ignorespaces%
899 \@oldnobreak%
900 \dump@pstartL@pc%
901 \ifnumberpstart%
902   \addtocounter{pstartL}{1}%
903 \fi
904 \parledgroup@beforenotes@save{L}%
905 %

```

Dump content of the optional argument.

```

906 \ifstrempy{#1}%
907   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
908   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
909 }
910 %

```

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

```

911 \newcommandx*{\pendR}[1][1]{%
912   \ifnumberingR \else%
913     \led@err@PendNotNumbered%
914   \fi%
915   \ifnumberedpar@ \else%
916     \led@err@PendNoPstart%
917   \fi%
918   \endgraf\global\num@linesR=\prevgraf\egroup%
919   \global\par@lineR=0%
920   \endgroup%
921   \ignorespaces%
922   \@oldnobreak%
923   \dump@pstartR@pc%
924   \ifnumberpstart%

```

```

925 \addtocounter{pstartR}{1}%
926 \fi%
927 \parledgroup@beforenotes@save{R}%
928 \ifstrepty{#1}%
929   {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
930   {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
931 }
932
933 %

```

\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.

```

934 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call{#1}}%
935 \gdef\at@every@pstart@call{}%
936 %

```

\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

```

937 \newif\ifprint@last@after@pendL%
938 \newif\ifprint@last@after@pendR%
939 %

```

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

```

940 \newbox\l@dleftbox
941 \newbox\l@drightbox
942
943 %

```

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

\countRline

```

944 \newcount\countLline
945 \countLline \z@
946 \newcount\countRline
947 \countRline \z@
948
949 %

```

\@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

```

950 \newcount\@donereallinesL
951 \newcount\@donetotallinesL
952 \newcount\@donereallinesR
953 \newcount\@donetotallinesR
954
955 %

```

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

```

956 \newcommand*\do@lineL{%
957   \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
958   \advance\countLline \@ne%
959   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
960     {\vbadness=10000%
961      \splittopskip=\z@%
962      \do@lineLhook%
963      \l@demptyd@ta%
964      \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}%
965        to\baselineskip}%
966     \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
967       \parledgroup@notes@startL}{}%
968     \unvbox\one@line \global\setbox\one@line=\lastbox%
969     \@writepageofparL%
970     \getline@numL%
971     \ifnum\@lock>\@ne%
972       \inserthangingsymboltrue%
973     \else%
974       \inserthangingsymbolfalse%
975     \fi%
976     \setbox\l@dleftbox%
977     \hb@xt@ \Lcolwidth{%
978       \ifl@dhiddenumber%
979       \global\l@dhiddenumberfalse%
980       \f@x@l@cks%
981     \else%
982       \affixline@num%
983     \fi%
984     \xifinlist{\the\l@dpscl}{\eled@sections@@}%
985     {\add@inserts\affixside@note}%
986     {\print@lineL}%
987   }%
988   \add@penaltiesL%
989   \global\advance\@donereallinesL\@ne%
990   \global\advance\@donetotallinesL\@ne%
991 \else%
992   \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
993   \global\advance\@donetotallinesL\@ne%
994 \fi%
995 }%

```

```

995
996
997 %

```

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

```

998 \def\print@lineL{%
999   \affixpstart@numL%
1000   \l@dld@ta %space kept for backward compatibility
1001   \add@inserts\affixside@note%
1002   \l@dlsn@te %space kept for backward compatibility
1003   {\ledllfill\hb@xt@ \Lcolwidth{%
1004     \do@insidelineLhook%
1005     \ifluatex%
1006       \texdir\l@luatextextdir@L%
1007     \fi%
1008     \new@lineL%
1009     \inserthangingsymbolL%
1010     \l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%
1011   \l@drsn@te}}
1012
1013 %

```

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

```

1014 \def\print@eledsectionL{%%
1015   \addtocounter{pstartL}{-1}%
1016   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
1017   \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}{%
1018   \numdef{\temp@}{\l@dpscL-1}%
1019   \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1020   \@eled@sectioningtrue%
1021   \bgroup%
1022     \ifluatex%
1023       \texdir\l@luatextextdir@L%
1024       \pardir\l@luatexpardir@L%
1025       \bodydir\l@luatexbodydir@L%
1026       \ifdefstring{\l@luatextextdir@L}{TRT}{\@RTLtrue}{%
1027     \fi%
1028     \csuse{eled@sectioning@the\l@dpscL}%
1029   \egroup%
1030   \@eled@sectioningfalse%
1031   \global\csundef{eled@sectioning@the\l@dpscL}%
1032   \if@RTL%
1033     \hspace{-3\paperwidth}%
1034     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1035   \else%
1036     \hspace{3\paperwidth}%
1037     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%

```

```

1038 \fi%
1039 \vskip\eledsection@correcting@skip%
1040 }
1041
1042 %

```

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

```

\dolineRhook
\doinsidelineLhook
\doinsidelineRhook
1043 \newcommand*\dolineLhook[1]{\gdef\do@lineLhook{#1}}%
1044 \newcommand*\dolineRhook[1]{\gdef\do@lineRhook{#1}}%
1045 \newcommand*\doinsidelineLhook[1]{\gdef\do@insidelineLhook{#1}}%
1046 \newcommand*\doinsidelineRhook[1]{\gdef\do@insidelineRhook{#1}}%
1047
1048 %

```

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

```

\dolineRhook
\do@insidelineLhook
\do@insidelineRhook
1049 \newcommand*\do@lineLhook{}
1050 \newcommand*\do@lineRhook{}
1051 \newcommand*\do@insidelineLhook{}
1052 \newcommand*\do@insidelineRhook{}
1053
1054 %

```

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

```

1055 \newcommand*\do@lineR{%
1056 \let\linenumrep\linenumrepR%
1057 \let\sublinenumrep\sublinenumrepR%
1058 \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscr}%
1059 \ledRcol@true%
1060 \advance\countRline \@ne%
1061 \ifvbox\namebox{\l@dRcolrawbox\the\l@dpscr}%
1062 {\vbadness=10000%
1063 \splittopskip=\z@%
1064 \do@lineRhook%
1065 \l@demptyd@ta%
1066 \global\setbox\one@lineR=\vsplit\namebox{\l@dRcolrawbox\the\l@dpscr}%
1067 to\baselineskip}%
1068 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
\parledgroup@notes@startR}{}%
1069 \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
1070 \@writepageofparR%
1071 \getline@numR%
1072 \ifnum\@lockR>\@ne%
1073 \inserthangingsymbolRtrue%
1074 \else%
1075 \inserthangingsymbolRfalse%

```

```

1076 \fi%
1077 \setbox\l@drightbox%
1078 \hb@xt@ \Rcolwidth{%
1079 \ifl@dhidenumber%
1080 \global\l@dhidenumberfalse%
1081 \f@x@l@cksR%
1082 \else%
1083 \affixline@numR%
1084 \fi%
1085 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1086 {\add@insertsR\affixside@noteR}%
1087 {\print@lineR}%
1088 }%
1089 \add@penaltiesR%
1090 \global\advance\@donereallinesR\@ne%
1091 \global\advance\@donetotallinesR\@ne%
1092 \else%
1093 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*\Rcolwidth}%
1094 \global\advance\@donetotallinesR\@ne%
1095 \fi%
1096 \ledRcol@false%
1097 }
1098
1099
1100 %

```

```

\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.

```

1101 \newcommand*{\getline@numR}{%
1102 \global\advance\absline@numR \@ne
1103 \do@actionsR
1104 \do@ballastR
1105 \ifledgroupnotesR\else
1106 \ifnumberline
1107 \ifsublines@
1108 \ifnum\sub@lockR<\tw@
1109 \global\advance\subline@numR \@ne
1110 \fi
1111 \else
1112 \ifnum\@lockR<\tw@
1113 \global\advance\line@numR \@ne
1114 \global\subline@numR \z@
1115 \fi
1116 \fi
1117 \fi

```



```

1118 \fi
1119 }
1120 \newcommand*{\getline@numL}{%
1121 \global\advance\absline@num \@ne
1122 \do@actions
1123 \do@ballast
1124 \ifledgroupnotesL\else
1125 \ifnumberline
1126 \ifsublines@
1127 \ifnum\sub@lock<\tw@
1128 \global\advance\subline@num \@ne
1129 \fi
1130 \else
1131 \ifnum\@lock<\tw@
1132 \global\advance\line@num \@ne
1133 \global\subline@num \z@
1134 \fi
1135 \fi
1136 \fi
1137 \fi
1138 }
1139
1140
1141 %

```

`\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.

```

1142 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1143 \begingroup
1144 \advance\absline@numR \@ne
1145 \ifnum\next@actionlineR=\absline@numR
1146 \ifnum\next@actionR>-1001
1147 \global\advance\ballast@count by -\c@ballast
1148 \fi
1149 \fi
1150 \endgroup}
1151 %

```

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

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

```

1152
1153 \newif\ifl@dskipversenumberR
1154 \newcommand*{\do@actions@fixedcodeR}{%
1155 \ifcase\l@dttempcnta%
1156 \or% % 1001
1157 \global\sublines@true

```

```

1158 \or% % 1002
1159 \global\sublines@false
1160 \or% % 1003
1161 \global\@lockR=\@ne
1162 \or% % 1004%
1163 \ifnum\@lockR=\tw@
1164 \global\@lockR=\thr@@
1165 \else
1166 \global\@lockR=\z@
1167 \fi
1168 \or% % 1005
1169 \global\sub@lockR=\@ne
1170 \or% % 1006
1171 \ifnum\sub@lockR=\tw@
1172 \global\sub@lockR=\thr@@
1173 \else
1174 \global\sub@lockR=\z@
1175 \fi
1176 \or% % 1007
1177 \l@dskipnumbertrue
1178 \or% % 1008
1179 \l@dskipversenumberRtrue%
1180 \or% % 1009
1181 \l@dhidenumbertrue%
1182 \else%
1183 \led@warn@BadAction
1184 \fi%
1185 }
1186
1187
1188 \newcommand*{\do@actionsR}{%
1189 \global\let\do@actions@nextR=\relax
1190 \@l@dttempcntb=\absline@numR
1191 \ifnum\@l@dttempcntb<\next@actionlineR\else
1192 \ifnum\next@actionR>-1001\relax
1193 \global\page@numR=\next@actionR
1194 \ifbypage@R
1195 \global\line@numR \z@ \global\subline@numR \z@
1196 \fi
1197 \else
1198 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1199 \@l@dttempcnta=-\next@actionR
1200 \advance\@l@dttempcnta by -5001\relax
1201 \ifsublines@
1202 \global\subline@numR=\@l@dttempcnta
1203 \else
1204 \global\line@numR=\@l@dttempcnta
1205 \fi
1206 \else
1207 \@l@dttempcnta=-\next@actionR

```

```

1208     \advance\@l@dttempcnta by -1000\relax
1209     \do@actions@fixedcodeR
1210     \fi
1211   \fi
1212   \ifx\actionlines@listR\empty
1213     \gdef\next@actionlineR{1000000}%
1214   \else
1215     \gl@p\actionlines@listR\to\next@actionlineR
1216     \gl@p\actions@listR\to\next@actionR
1217     \global\let\do@actions@nextR=\do@actionsR
1218   \fi
1219 \fi
1220 \do@actions@nextR}
1221
1222 %

```

VIII.4 Line number printing

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

```

\ch@cksub@l@ckR
\ch@ck@l@ckR
\fx@l@cksR
\affixline@numR
1223
1224 \newcommand*{\l@dcalcnm}[3]{%
1225   \ifnum #1 > #2\relax
1226     \@l@dttempcnta = #1\relax
1227     \advance\@l@dttempcnta by -#2\relax
1228     \divide\@l@dttempcnta by #3\relax
1229     \multiply\@l@dttempcnta by #3\relax
1230     \advance\@l@dttempcnta by #2\relax
1231   \else
1232     \@l@dttempcnta=#2\relax
1233   \fi}
1234
1235 \newcommand*{\ch@cksub@l@ckR}{%
1236   \ifcase\sub@lockR
1237   \or
1238     \ifnum\sublock@disp=\@ne
1239       \@l@dttempcntb \z@ \@l@dttempcnta \@ne
1240     \fi
1241   \or
1242     \ifnum\sublock@disp=\tw@
1243     \else
1244       \@l@dttempcntb \z@ \@l@dttempcnta \@ne
1245     \fi
1246   \or
1247     \ifnum\sublock@disp=\z@
1248       \@l@dttempcntb \z@ \@l@dttempcnta \@ne
1249     \fi
1250   \fi}
1251

```

```

1252 \newcommand*{\check@l@ckR}{%
1253   \ifcase\@lockR
1254   \or
1255     \ifnum\lock@disp=\@ne
1256       \@l@tempcntb \z@ \l@tempcnta \@ne
1257     \fi
1258   \or
1259     \ifnum\lock@disp=\tw@
1260     \else
1261       \@l@tempcntb \z@ \l@tempcnta \@ne
1262     \fi
1263   \or
1264     \ifnum\lock@disp=\z@
1265       \@l@tempcntb \z@ \l@tempcnta \@ne
1266     \fi
1267   \fi}
1268
1269 \newcommand*{\f@x@l@cksR}{%
1270   \ifcase\@lockR
1271   \or
1272     \global\@lockR \tw@
1273   \or \or
1274     \global\@lockR \z@
1275   \fi
1276   \ifcase\sub@lockR
1277   \or
1278     \global\sub@lockR \tw@
1279   \or \or
1280     \global\sub@lockR \z@
1281   \fi}
1282
1283
1284 \newcommand*{\affixline@numR}{%
1285   \ifledgroupnotesR\else\ifnumberline
1286   \ifl@dskipnumber
1287     \global\l@dskipnumberfalse
1288   \else
1289     \ifsublines@
1290       \@l@tempcntb=\subline@numR
1291       \l@dcalcnun{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR
1292     }%
1293   \ch@cksub@lockR
1294   \else
1295     \@l@tempcntb=\line@numR
1296     \ifx\linenumberlist\empty
1297       \l@dcalcnun{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1298     \else
1299       \@l@tempcnta=\line@numR
1300       \edef\rem@inder{\linenumberlist,\number\line@numR,}%
1301       \edef\sc@n@list{\def\noexpand\sc@n@list

```

```

1301      ###1,\number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1302      \sc@n@list\expandafter\sc@n@list\rem@inder|%
1303      \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1304      \fi
1305      \ch@ck@l@ckR
1306      \fi
1307      \ifnum\@l@tempcnta=\@l@tempcntb
1308      \ifl@dskipversenumberR\else
1309      \if@twocolumn
1310      \if@firstcolumn
1311      \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1312      \else
1313      \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1314      \fi
1315      \else
1316      \@l@tempcntb=\line@marginR
1317      \ifnum\@l@tempcntb>\@ne
1318      \advance\@l@tempcntb by\page@numR
1319      \fi
1320      \ifodd\@l@tempcntb
1321      \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1322      \else
1323      \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1324      \fi
1325      \fi
1326      \fi
1327      \fi
1328      \f@x@l@cksR
1329      \fi
1330      \fi
1331      \fi}
1332      %

```

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@numL333
\affixpstart@numR334 \newcommand*{\affixpstart@numL}{%
  \leftpstartnumR335 \ifsidepstartnum
  \rightpstartnumR336 \if@twocolumn
  \leftpstartnumL337 \if@firstcolumn
  \rightpstartnumL338 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
  \ifpstartnumR

```

```

1339 \else
1340 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1341 \fi
1342 \else
1343 \l@dtempcntb=\line@margin
1344 \ifnum\l@dtempcntb>\@ne
1345 \advance\l@dtempcntb \page@num
1346 \fi
1347 \ifodd\l@dtempcntb
1348 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}}%
1349 \else
1350 \gdef\l@dld@ta{\llap{\leftpstartnumL}}}%
1351 \fi
1352 \fi
1353 \fi
1354 }
1355 \newcommand*\affixpstart@numR{%
1356 \ifsidepstartnum
1357 \if@twocolumn
1358 \if@firstcolumn
1359 \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1360 \else
1361 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1362 \fi
1363 \else
1364 \l@dtempcntb=\line@marginR
1365 \ifnum\l@dtempcntb>\@ne
1366 \advance\l@dtempcntb \page@numR
1367 \fi
1368 \ifodd\l@dtempcntb
1369 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}}%
1370 \else
1371 \gdef\l@dld@ta{\llap{\leftpstartnumR}}}%
1372 \fi
1373 \fi
1374 \fi
1375 }
1376
1377 \newcommand*\leftpstartnumL{%
1378 \ifpstartnum
1379 \thepstartL
1380 \kern\linenumsep\global\pstartnumfalse\fi
1381 }
1382 \newcommand*\rightpstartnumL{%
1383 \ifpstartnum\kern\linenumsep
1384 \thepstartL
1385 \global\pstartnumfalse\fi
1386 }
1387 \newif\ifpstartnumR
1388 \pstartnumRtrue

```

```

1389 \newcommand*{\leftpstartnumR}{
1390 \ifpstartnumR
1391 \thepstartR
1392 \kern\linenumsep\global\pstartnumRfalse\fi
1393 }
1394 \newcommand*{\rightpstartnumR}{
1395 \ifpstartnumR\kern\linenumsep
1396 \thepstartR
1397 \global\pstartnumRfalse\fi
1398 }
1399 %

```

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.

```

1400 \list@create{\inserts@listR}
1401 %

```

`\add@insertsR` The right text version.

```

\add@inserts@nextR
1402 \newcommand*{\add@insertsR}{%
1403 \global\let\add@inserts@nextR=\relax
1404 \ifx\inserts@listR\empty \else
1405 \ifx\next@insertR\empty
1406 \ifx\insertlines@listR\empty
1407 \global\noteschanged@true
1408 \gdef\next@insertR{100000}%
1409 \else
1410 \gl@p\insertlines@listR\to\next@insertR
1411 \fi
1412 \fi
1413 \ifnum\next@insertR=\absline@numR
1414 \gl@p\inserts@listR\to\@insertR
1415 \@insertR
1416 \global\let\@insertR=\undefined
1417 \global\let\next@insertR=\empty
1418 \global\let\add@inserts@nextR=\add@insertsR
1419 \fi
1420 \fi
1421 \add@inserts@nextR}
1422 %
1423 %

```

VIII.7 Penalties

`\add@penaltiesL` `\add@penaltiesL` is the last macro used by `\do@lineL`. It adds up the club, widow, and interline penalties, and puts a single penalty of the appropriate size back into the para-

graph; 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@tempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast`. Finally, the penalty is checked to see that it does not go below -10000 .

```
\newcommand*{\add@penaltiesR}{\@l@tempcnta=\ballast@count
\ifnum\num@linesR>\@ne
\global\advance\par@lineR \@ne
\ifnum\par@lineR=\@ne
\advance\@l@tempcnta by \clubpenalty
\fi
\@l@tempcntb=\par@lineR \advance\@l@tempcntb \@ne
\ifnum\@l@tempcntb=num@linesR
\advance\@l@tempcnta by \widowpenalty
\fi
\ifnum\par@lineR<\num@linesR
\advance\@l@tempcnta by \interlinepenalty
\fi
\fi
\ifnum\@l@tempcnta=\z@
\relax
\else
\ifnum\@l@tempcnta>-10000
\penalty\@l@tempcnta
\else
\penalty -10000
\fi
\fi}
```

This is for a single chunk. However, as we are probably dealing with several chunks at a time, the above is not really relevant. 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.

```
1424 \newcommand*{\add@penaltiesL}{\}
1425 \newcommand*{\add@penaltiesR}{\}
1426
1427 %
```

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.


```

1428 \newcommand*{\flush@notesR}{%
1429   \@xloop
1430   \ifx\inserts@listR\empty \else
1431     \gl@p\inserts@listR\to\@insertR
1432     \@insertR
1433     \global\let\@insertR=\undefined
1434   \repeat}
1435
1436 %

```

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

```

```

1437 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1438   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1439   \ifl@d@pnum #1\fullstop\fi
1440   \ifl@d@plinum \linenumr@p{#2}\@Rlineflag\else \symplinum\fi
1441   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1442   \ifl@d@dash \endashchar\fi
1443   \ifl@d@pnum #4\fullstop\fi
1444   \ifl@d@elin \linenumr@p{#5}\@Rlineflag\fi
1445   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1446 \endgroup}
1447
1448 %
1449 %

```

IX.2 Footnotes output specific to `\Pages`

`\print@Xnotes@forpages` 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 `\correct@Xfootins@box` `\print@notesX@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed⁶:

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

- 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:

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

by `\correct@Xfootins@box` (or `\correct@footinsX@box` for familiar notes).

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

```
1450 \newcommand\print@Xnotes@forpages[1]{%
1451 %
```

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

```
1452 \ifcempty{Xonlyside@#1}{%
1453 \csuse{#1footstart}{#1}%
1454 \csuse{#1footgroup}{#1}%
1455 }%
1456 %
```

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

```
1457 {%
1458 \ifboolexpr{%
1459 ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\ifnumodd{\c@page
}})%
1460 or%
1461 (test {\ifcsstring{Xonlyside@#1}{R}} and test{\ifnumodd{\c@page}})%
1462 }%
1463 %
```

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

```
1464 {%
1465 \correct@Xfootins@box{#1}%
1466 \csuse{#1footstart}{#1}%
1467 \csuse{#1footgroup}{#1}%
1468 %
```

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

```
1469 \global\count\csuse{#1footins}=0%
1470 \global\skip\csuse{#1footins}=0pt%
1471 %
```

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

```
1472 \csuse{Xnotefontsize@#1}%
1473 \global\advance\dimen\csuse{#1footins} by -\baselineskip%
1474 %
```

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

```
1475 \global\boolfalse{keepforXside@#1}%
1476 }%
1477 %
```

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.

```
1478 {%
1479 \global\booltrue{keepforXside@#1}%
1480 %
```

Then restore expected rooms for notes on the next page.

```
1481 \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1482 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
1483 %
```

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

```
1484 \bgroup%
1485 \csuse{Xnotefontsize@#1}%
1486 \global\advance\dimen\csuse{#1footins} by \baselineskip%
1487 \egroup%
1488 %
```

```
1489 % End of \protect\cs{print@Xnotes@forpages}.
1490 }%
1491 }%
1492 }%
1493 %
```

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

```
1494 \newcommand{\correct@Xfootins@box}[1]{%
1495 %
```

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.

```
1496 \ifbool{keepforXside@#1}{%
1497 %
```

Some setting needed to do the right splitting.

```
1498 \csuse{Xnotefontsize@#1}%
1499 \splittopskip=0pt%
1500 %
```

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

```
1501 \global\setbox\csuse{#1footins}=\vbox{%
1502 \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1503 \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1504 \unvbox\csuse{#1footins}%
1505 }%
1506 %
```

End of the macro.

```

1507 }{}%
1508 }%
1509 %

```

And now, the same for familiar footnotes.

```

1510 \newcommand\print@notesX@forpages[1]{%
1511   \ifcseempty{onlysideX@#1}{%
1512     \csuse{footstart#1}{#1}%
1513     \csuse{footgroup#1}{#1}%
1514   }%
1515   {%
1516     \ifboolexpr{%
1517       ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page
1518     }})%
1519       or%
1520       (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}}))%
1521     }%
1522     \correct@footinsX@box{#1}%
1523     \csuse{footstart#1}{#1}%
1524     \csuse{footgroup#1}{#1}%
1525     \global\count\csuse{footins#1}=0%
1526     \global\skip\csuse{footins#1}=0pt%
1527     \csuse{notefontsizeX@#1}%
1528     \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1529     \global\boolfalse{keepforsideX@#1}%
1530   }%
1531   {%
1532     \global\booltrue{keepforsideX@#1}%
1533     \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1534     \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1535     \bgroup%
1536     \csuse{notefontsizeX@#1}%
1537     \global\advance\dimen\csuse{footins#1} by \baselineskip%
1538     \egroup%
1539   }%
1540 }%
1541 }%
1542 \newcommand{\correct@footinsX@box}[1]{%
1543   \ifbool{keepforsideX@#1}{%
1544     \csuse{notefontsizeX@#1}%
1545     \splittopskip=0pt%
1546     \global\setbox\csuse{footins#1}=\vbox{%
1547       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1548       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1549       \unvbox\csuse{footins#1}%
1550     }%
1551   }{}%
1552 }%
1553 %

```

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.

```
1554 \list@create{\labelref@listR}
1555
1556 %
```

`\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`.

```
1557 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1558   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1559     \led@warn@DuplicateLabel{#4}%
1560   \fi
1561   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\@Rlineflag|#3|#4}%
1562   \ignorespaces}
1563 \AtBeginDocument{%
1564   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1565   }
1566
1567 %
```

`\@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*
1568 \WithSuffix\newcommand\sidenotemargin*[1]{%
1569   \l@dgetsidenote@margin{#1}
1570   \global\sidenote@marginR=\@l@dttempcntb
1571   \global\sidenote@margin=\@l@dttempcntb
1572 }
1573 \newcount\sidenote@marginR
1574 \global\sidenote@margin=\@ne
1575
1576 %
```

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

```

1577 \newcommand*{\affixside@noteR}{%
1578   \def\sidenotecontent@{%
1579     \numgdef{\itemcount@}{0}%
1580     \def\do##1{%
1581       \ifnumequal{\itemcount@}{0}%
1582         {%
1583           \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
1584           {\appto\sidenotecontent@{\sidenotessep ##1}%
1585           }%
1586           \numgdef{\itemcount@}{\itemcount@+1}%
1587         }%
1588         \dolistloop{\l@dcstetext}%
1589         \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1590       \gdef\@templ@d{%
1591         \gdef\@templ@n{\l@dcstetext\l@dcstetext@l\l@dcstetext@r}%
1592         \ifx\@templ@d\@templ@n \else%
1593           \if@twocolumn%
1594             \if@firstcolumn%
1595               \setl@dlp@rbox{##1}{\sidenotecontent@}%
1596             \else%
1597               \setl@drp@rbox{\sidenotecontent@}%
1598             \fi%
1599           \else%
1600             \l@dttempcntb=\sidenote@marginR%
1601             \ifnum\l@dttempcntb>\@ne%
1602               \advance\l@dttempcntb by\page@numR%
1603             \fi%
1604             \ifodd\l@dttempcntb%
1605               \setl@drp@rbox{\sidenotecontent@}%
1606             \gdef\sidenotecontent@{%
1607               \numdef{\itemcount@}{0}%
1608               \dolistloop{\l@dcstetext@l}%
1609               \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1610               \setl@dlp@rbox{\sidenotecontent@}%
1611             \else%
1612               \setl@dlp@rbox{\sidenotecontent@}%
1613             \gdef\sidenotecontent@{%
1614               \numdef{\itemcount@}{0}%
1615               \dolistloop{\l@dcstetext@r}%
1616               \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1617               \setl@drp@rbox{\sidenotecontent@}%
1618             \fi%
1619           \fi%
1620         \fi%
1621       }
1622     }%
1623   }%

```

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.

```
\inserthangingsymbolL24 \newif\ifinserthangingsymbolR
\inserthangingsymbolR25 \newcommand{\inserthangingsymbolL}{%
1626   \ifinserthangingsymbol%
1627     \ifinstanzaL%
1628       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1629         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1630       \@hangingsymbol%
1631     \fi%
1632   \fi%
1633 }%
1634 \newcommand{\inserthangingsymbolR}{%
1635   \ifinserthangingsymbolR%
1636     \ifinstanzaR%
1637       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1638         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1639       \@hangingsymbol%
1640     \fi%
1641   \fi%
1642 }%
1643 %
```

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.

```
1644 \chardef\next=\catcode`\&
1645 \catcode`\&=\active
1646
1647 %
```

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

```
1648 \newenvironment{astanza}[1][\&]{%
1649   \catcode`\&=\active
1650   \global\stanza@count\@ne\stanza@modulo\@ne
```



```

1651 \ifnum\usernamecount{sza@0@}=\z@
1652   \let\stanza@hang\relax
1653   \let\endlock\relax
1654 \else
1655   \rightskip\z@ plus 1fil\relax
1656 \fi
1657 \ifnum\usernamecount{szp@0@}=\z@
1658   \let\sza@penalty\relax
1659 \fi
1660 \def&{%
1661   \endlock\mbox{}%
1662   \sza@penalty
1663   \global\advance\stanza@count\@ne
1664   \@astanza@line}%
1665 \def\&{\@stopastanza}%
1666 \pstart[#1]%
1667 \@astanza@line
1668 }{}
1669
1670 %

```

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

```

1671 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1672   \endlock\mbox{}%
1673   \pend[#1]%
1674 }%
1675 %

```

\@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.

```

1676 \newcommand*{\@astanza@line}{%
1677   \ifnum\value{stanzaindentrepetition}=0
1678     \parindent=\csname sza@\number\stanza@count
1679       @\endcsname\stanzaindentbase
1680   \else
1681     \parindent=\csname sza@\number\stanza@modulo
1682       @\endcsname\stanzaindentbase
1683     \managestanza@modulo
1684   \fi
1685   \par
1686   \stanza@hang%\mbox{}%
1687   \ignorespaces}
1688
1689 %

```

Lastly reset the modified category codes.

```

1690 \catcode`\&=\next
1691
1692 %

```

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

```

\thestanzaR
1693 \newcounter{stanzaL}
1694 \newcounter{stanzaR}
1695 \renewcommand{\thestanzaL}{%
1696   \textbf{\arabic{stanzaL}}%
1697 }
1698 \renewcommand{\thestanzaR}{%
1699   \textbf{\arabic{stanzaR}}%
1700 }
1701 %
1702 %

```

XIV Naming macros

The \LaTeX kernel provides $\backslash\@namedef$ and $\backslash\@namuse$ 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
1703 \providecommand*\newnamebox}[1]{%
1704   \expandafter\newbox\csname #1\endcsname}
1705 \providecommand*\setnamebox}[1]{%
1706   \expandafter\setbox\csname #1\endcsname}
1707 \providecommand*\unhnamebox}[1]{%
1708   \expandafter\unhbox\csname #1\endcsname}
1709 \providecommand*\unvnamebox}[1]{%
1710   \expandafter\unvbox\csname #1\endcsname}
1711 \providecommand*\namebox}[1]{%
1712   \csname #1\endcsname}
1713
1714 %

```

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

```

\usenamecount
1715 \providecommand*\newnamecount}[1]{%
1716   \expandafter\newcount\csname #1\endcsname}
1717 \providecommand*\usenamecount}[1]{%
1718   \csname #1\endcsname}
1719
1720 %

```

XV Fixing babel and polyglossia

With parallel texts there is the possibility that the two sides might use different languages via babel. On the other hand, nor babel nor polyglossia might not be called at all (even though it might be already built into the format).

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

```
\ifl@dusedbabel A flag for checking if babel has been used as a package.
\l@dusedbabelfalse
\l@dusedbabeltrue1721
1722 %
```

```
\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 `\`.

```
1723 \patchcmd{\bbl@set@language}%
1724 {\select@language{\language}}%
1725 {\edef\language{#1}\select@language{\language}}%
1726 {}%
1727 {}%
1728
1729 %
```

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
1730 \newcommand*{\l@duselanguage}[1]{\{
1731 \gdef\theledlanguageL{\{
1732 \gdef\theledlanguageR{\{
1733
1734 %
```

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

```

1735 \AtBeginDocument{%
1736   \ifundefined{xpg@main@language}{%
1737     \ifundefined{bbl@main@language}{%
1738       %

```

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

```

1739   \l@dusedbabelfalse
1740 }{%
1741 %

```

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.

```

1742   \l@dusedbabeltrue
1743   \let\l@doldselectlanguage\selectlanguage
1744   \let\l@doldbbl@set@language\bbl@set@language
1745   \renewcommand{\selectlanguage}[1]{%
1746     \l@doldselectlanguage{#1}%
1747     \ifledRcol \gdef\theledlanguageR{#1}%
1748     \else      \gdef\theledlanguageL{#1}%
1749     \fi}
1750 %

```

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

```

1751   \renewcommand*{\l@duselanguage}[1]{%
1752     \l@doldselectlanguage{#1}}
1753 %

```

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

```

1754   \gdef\theledlanguageL{\bbl@main@language}%
1755   \gdef\theledlanguageR{\bbl@main@language}%
1756 }%
1757 }
1758 %

```

If use polyglossia

```

1759 {   \let\old@otherlanguage\otherlanguage%
1760     \renewcommand{\otherlanguage}[2][ ]{%
1761       \selectlanguage{#1}{#2}%
1762       \ifledRcol \gdef\theledlanguageR{#2}%
1763       \else      \gdef\theledlanguageL{#2}%
1764       \fi}%
1765     \let\l@duselanguage\select@language%
1766     \gdef\theledlanguageL{\xpg@main@language}%
1767     \gdef\theledlanguageR{\xpg@main@language}%
1768 %

```

That is it.

```

1769 }}
1770 %

```

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.

```

1771 \newcount\l@dc@maxchunks
1772 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1773 \maxchunks{5120}
1774
1775 %

```

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

```

\l@dnumpstartsR
1776 \newcount\l@dnumpstartsR
1777
1778 %

```

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

```

\l@pscl
1779 \newcount\l@dpscl
1780 \newcount\l@dpscl
1781
1782 %

```

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

```

1783 \newcommand*{\l@dsetuprawboxes}{%
1784 \l@l@tempcntb=\l@dc@maxchunks
1785 \loop\ifnum\l@l@tempcntb>\z@
1786 \newnamebox{\l@dLcolrawbox\the\l@l@tempcntb}
1787 \newnamebox{\l@dRcolrawbox\the\l@l@tempcntb}
1788 \advance\l@l@tempcntb \m@ne
1789 \repeat}
1790
1791 %

```

`\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.

```

1792 \newcommand*{\l@dssetupmaxlinecounts}{%
1793   \l@dttempcntb=\l@dc@maxchunks
1794   \loop\ifnum\l@dttempcntb>\z@
1795     \newnamecount{\l@dsmaxlinesinpar\the\l@dttempcntb}
1796     \advance\l@dttempcntb \m@ne
1797   \repeat}
1798 \newcommand*{\l@dzeromaxlinecounts}{%
1799   \begingroup
1800   \l@dttempcntb=\l@dc@maxchunks
1801   \loop\ifnum\l@dttempcntb>\z@
1802     \global\usenamecount{\l@dsmaxlinesinpar\the\l@dttempcntb}=\z@
1803     \advance\l@dttempcntb \m@ne
1804   \repeat
1805   \endgroup}
1806
1807 %

```

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

```

1808 \AtBeginDocument{%
1809   \l@dssetuprawboxes
1810   \l@dssetupmaxlinecounts
1811   \l@dzeromaxlinecounts
1812   \l@dnumpstartsL=\z@
1813   \l@dnumpstartsR=\z@
1814   \l@dpscL=\z@
1815   \l@dpscR=\z@}
1816
1817 %

```

XVII Checking text to be processed

```

\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue
\@pstartsfalse
\check@pstarts
1818 \newif\if@pstarts
1819 \newcommand*{\check@pstarts}{%
1820   \@pstartsfalse
1821   \ifnum\l@dnumpstartsL>\l@dpscL
1822     \@pstartstrue
1823   \else
1824     \ifnum\l@dnumpstartsR>\l@dpscR
1825       \@pstartstrue
1826     \fi
1827   \fi
1828 }
1829
1830 %

```

`\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
1831 \newif\ifaraw@text
1832 \newcommand*\checkraw@text{%
1833   \araw@textfalse
1834   \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}
1835     \araw@texttrue
1836   \else
1837     \ifvbox\namebox{1@dRcolrawbox\the\1@dpscR}
1838       \araw@texttrue
1839     \fi
1840   \fi
1841 }
1842 %
1843 %

```

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

```

1844 \newcommand*\@writelinesinparL{%
1845   \edef\next{%
1846     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1847   \next
1848   \global\@donereallinesL \z@}
1849 \newcommand*\@writelinesinparR{%
1850   \edef\next{%
1851     \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1852   \next
1853   \global\@donereallinesR \z@}
1854 %
1855 %

```

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

```

\@writepageofparR
1856 \newcommand*\@writepageofparL[0]{%
1857   \ifnum\@donereallinesL=\z@%
1858     \edef\next{%
1859       \write\linenum@out{\string\@pstart{\the\1@dpscL}{\the\c@page}{\the\
1860       numpagelinesL}}%
1861     }%
1862     \next%
1863   \fi%
1864 }%
1865 \newcommand*\@writepageofparR[0]{%
1866   \ifnum\@donereallinesR=\z@%
1867     \edef\next{%
1868       \write\linenum@outR{\string\@pstartR{\the\1@dpscR}{\the\c@page}{\the\
1869       numpagelinesR}}%

```

```

1868 }%
1869 \next%
1870 \fi%
1871 }%
1872 %

```

XVIII Parallel columns

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

```

\@eledsectionR
1873 \newsavebox{\@eledsectionL}%
1874 \newsavebox{\@eledsectionR}%
1875 %

```

`\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.

```

1876 \newcommand*{\Columns}{%
1877   \ifl@dpairing%
1878     \led@err@Columns@InsideEnv%
1879   \fi%
1880   \l@dprintingcolumnstrue%
1881   \eledsection@correcting@skip=-\baselineskip% Correction for sections'
titles
1882   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1883     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1884   \fi
1885 %

```

Start a group and zero counters, etc.

```

1886 \begingroup
1887   \l@dzeropenalties
1888   \endgraf\global\num@lines=\prevgraf
1889   \global\num@linesR=\prevgraf
1890   \global\par@line=\z@
1891   \global\par@lineR=\z@
1892   \global\l@dpscL=\z@
1893   \global\l@dpscR=\z@
1894 %

```

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

```

1895 \check@pstarts
1896 \loop\if@pstarts
1897   \global\pstartnumtrue
1898   \global\pstartnumRtrue
1899 %

```


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.

```

1900 \global\advance\l@dpscL \@ne
1901 \global\advance\l@dpscR \@ne
1902 \restore@pstartL@pc%
1903 \restore@pstartR@pc%
1904 %

```

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

```

1905 \Columns@print@before@pstart%
1906 %

```

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

```

1907 \checkraw@text
1908 { \loop\ifaraw@text
1909 %

```

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

```

1910 \l@duselanguage{\theledlanguageL}%
1911 \do@lineL
1912 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1913 {%
1914 \ifdefstring{\@eledsectmark}{L}%
1915 {\csuse{eled@sectmark@\the\l@dpscL}%
1916 }}%
1917 \global\csundef{eled@sectmark@\the\l@dpscL}%
1918 \savebox{\@eledsectionL}{\parbox[t]{}[t]{\Lcolwidth}{\vbox
1919 {} \print@eledsectionL}}% \vbox{}-> prevent alignment troubles with RTL
1920 language
1921 }%
1922 {}%
1923 \l@duselanguage{\theledlanguageR}%
1924 \do@lineR
1925 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1926 {%
1927 \ifdefstring{\@eledsectmark}{R}%
1928 {\csuse{eled@sectmark@\the\l@dpscR R}%
1929 }}%
1930 \global\csundef{eled@sectmark@\the\l@dpscR R}%
1931 \savebox{\@eledsectionR}{\parbox[t]{}[t]{\Rcolwidth}{\vbox
1932 {} \print@eledsectionR}}% \vbox{}-> prevent alignment troubles with RTL
1933 language
1934 }%
1935 {}%
1936 \hb@xt@ \hsize{%
1937 \ifdefstring{\columns@position}{L}{\hfill}%
1938 \unhbox\l@dleftbox%
1939 \ifhbox{\@eledsectionL%

```

```

1935         \usebox{\@eledsectionL}%
1936         \fi%
1937         \print@columnseparator%
1938         \unhbox\l@drighthbox%
1939         \ifhbox\@eledsectionR%
1940         \usebox{\@eledsectionR}%
1941         \fi%
1942         \ifdefstring{\columns@position}{R}{\hfill}%
1943     }%
1944     \checkraw@text
1945     \checkverseL
1946     \checkverseR
1947     \checkpb@columns
1948     \repeat}
1949 %

```

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.

```

1950     \@writelinesinparL
1951     \@writelinesinparR
1952     \check@pstarts
1953     \ifbypstart@%
1954         \write\linenum@out{\string\@set[1]}
1955         \resetprevline@
1956     \fi
1957     \ifbypstart@R
1958         \write\linenum@outR{\string\@set[1]}
1959         \resetprevline@
1960     \fi
1961     \Columns@print@after@pend%
1962     \repeat
1963 %

```

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.

```

1964     \flush@notes
1965     \flush@notesR
1966     \endgroup
1967 %

1968 \global\l@dpscL=\z@
1969 \global\l@dpscR=\z@
1970 \global\l@dnpstartsL=\z@
1971 \global\l@dnpstartsR=\z@
1972 \l@dprintingcolumnsfalse%
1973 \ignorespaces
1974 \global\instanzaLfalse
1975 \global\instanzaRfalse}

```

```
1976
1977 %
```

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

```
1978 \def\print@columnseparator{%
1979   \ifdim\beforecolumnseparator<0pt%
1980     \hfill%
1981   \else%
1982     \hspace{\beforecolumnseparator}%
1983   \fi%
1984   \columnseparator%
1985   \ifdim\aftercolumnseparator<0pt%
1986     \hfill%
1987   \else%
1988     \hspace{\beforecolumnseparator}%
1989   \fi%
1990 }%
1991 %
```

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

```
1992
1993 \newcommand{\checkpb@columns}{%
1994   \newif\if@pb
1995   \newif\if@nopb
1996   \IfStrEq{\led@pb@setting}{before}{
1997     \numdef{\next@absline}{\the\absline@num+1}%
1998     \numdef{\next@abslineR}{\the\absline@numR+1}%
1999     \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
2000     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
2001       \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
2002       \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
2003         }{}
2004     \IfStrEq{\led@pb@setting}{after}{
2005       \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
2006       \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
2007       \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
2008       \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
2009         }{}
2010     \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
2011     \if@pb\pagebreak[4]\fi
2012   }
2013 }
```

\columnseparator The separator between line pairs in parallel columns is in the form of a vertical
\columnrulewidth 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).

```

2014 \newcommand*{\columnseparator}{%
2015   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}}
2016 \newdimen\columnrulewidth
2017   \columnrulewidth=\z@
2018
2019 %

```

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

```

\columns@position
2020 \newcommand*{\columnspan}[1]{%
2021   \xdef\columns@position{#1}%
2022 }%
2023 \xdef\columns@position{R}%
2024 %

```

`\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`.

```

2025 \newlength{\beforecolumnseparator}%
2026 \setlength{\beforecolumnseparator}{-2pt}%
2027
2028 \newlength{\aftercolumnseparator}%
2029 \setlength{\aftercolumnseparator}{-2pt}%
2030
2031 %

```

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

```

2032 \newcommand{\setwidthliketwocolumns@L}{%
2033   % Temporary dimension, initially equal to the standard hsize, i.e. text
2034   width
2035   \begin{macrocode}
2036   \newdimen\temp%
2037   \temp=\hsize%
2038 }%

```

Hsize : Left + Right width

```

2038 \hsize=\Lcolwidth%
2039 \advance\hsize\Rcolwidth%
2040 %

```

Now, calculating the remaining space

```
2041 \advance\temp-\hsize%
2042 %
```

And multiply the hsize by 2/3 of this space

```
2043 \multiply\temp by 2%
2044 \divide\temp by 3%
2045 \advance\hsize\temp%
2046 }%
2047
2048 \newcommand{\setpositionliketwocolumns@L}{%
2049 \renewcommand{\ledrlfill}{\hfill}%
2050 }%
2051
2052 \newcommand{\setnotespositionliketwocolumns@L}{%
2053 }%
2054
2055
2056 %
```

```
2057 \newcommand{\setwidthliketwocolumns@C}{%
2058 % Temporary dimension, initially equal to the standard hsize, i.e. text
2059 % width
2060 %
```

```
2060 \newdimen\temp%
2061 \temp=\hsize%
2062 % Hsize : Left + Right width
2063 %
```

```
2064 \hsize=\Lcolwidth%
2065 \advance\hsize\Rcolwidth%
2066 % Now, calculating the remaining space
2067 %
```

```
2068 \advance\temp-\hsize%
2069 %
```

And multiply the hsize by 1/2 of this space

```
2070 \divide\temp by 2%
2071 \advance\hsize\temp%
2072 }%
2073
2074 \newcommand{\setpositionliketwocolumns@C}{%
2075 \doinsidelinehook{\hfill}%
2076 \renewcommand{\ledrlfill}{\hfill}%
2077 }%
2078
2079 \newcommand{\setnotespositionliketwocolumns@C}{%
```

```

2080 \newdimen\temp%
2081 \newdimen\tempa%
2082 \temp=\hsize%
2083 \tempa=\Lcolwidth%
2084 \advance\tempa\Rcolwidth%
2085 \advance\temp-\tempa%
2086 \divide\temp by 2%
2087 \leftskip=\temp%
2088 \rightskip=-\temp%
2089 }%
2090
2091 \newcommand{\setwidthliketwocolumns@R}{%
2092 %

```

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

```

2093 \newdimen\temp%
2094 \temp=\hsize%
2095 %

```

Hsize : Left + Right width

```

2096 \hsize=\Lcolwidth%
2097 \advance\hsize\Rcolwidth%
2098 %

```

Now, calculating the remaining space

```

2099 \advance\temp-\hsize%
2100 %

```

And multiply the hsize by 2/3 of this space

```

2101 \multiply\temp by 2%
2102 \divide\temp by 3%
2103 \advance\hsize\temp%
2104 }%
2105
2106 \newcommand{\setpositionliketwocolumns@R}{%
2107 \doinsidelinehook{\hfill}%
2108 }%
2109
2110 \newcommand{\setnotespositionliketwocolumns@R}{%
2111 \newdimen\temp%
2112 \newdimen\tempa%
2113 \temp=\hsize%
2114 \tempa=\Lcolwidth%
2115 \advance\tempa\Rcolwidth%
2116 \advance\temp-\tempa%
2117 \divide\temp by 2%
2118 \leftskip=\temp%
2119 \rightskip=-\temp%
2120 }%

```

2121
2122 %

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

```
2123 \newcommand{\Columns@print@before@pstart}{%
2124   \ifboolexpr{%
2125     test{\ifcsstring{before@pstartL@the\l@dpscL}{\at@every@pstart}}%
2126     and test {\ifcsstring{before@pstartR@the\l@dpscR}{\at@every@pstart}}%
2127     and test {\ifdefempty{\at@every@pstart}}}%
2128     {%
2129     {%
2130       \hb@xt@ \hsize{%
2131         \ifdefstring{\columns@position}{L}{\hfill }%
2132         \par\parbox[t] [] [t]{\Lcolwidth}{%
2133           \csuse{before@pstartL@the\l@dpscL}%
2134         }%
2135         \print@columnseparator%
2136         \parbox[t] [] [t]{\Rcolwidth}{%
2137           \initnumbering@sectcountR%
2138           \csuse{before@pstartR@the\l@dpscR}%
2139         }%
2140         \ifdefstring{\columns@position}{R}{\hfill}%
2141       }%
2142     }%
2143     \global\csundef{before@pstartL@the\l@dpscL}%
2144     \global\csundef{before@pstartR@the\l@dpscR}%
2145   }%
2146 \newcommand{\Columns@print@after@pend}{%
2147   \ifboolexpr{%
2148     test{\ifcsstring{after@pendL@the\l@dpscL}{\at@every@pend}}%
2149     and test {\ifcsstring{after@pendR@the\l@dpscR}{\at@every@pend}}%
2150     and test {\ifdefempty{\at@every@pend}}}%
2151     {%
2152     {%
2153       \hb@xt@ \hsize{%
2154         \ifdefstring{\columns@position}{L}{\hfill }%
2155         \parbox[t] [] [t]{\Lcolwidth}{%
2156           \csuse{after@pendL@the\l@dpscL}%
2157         }%
2158         \print@columnseparator%
2159         \parbox[t] [] [t]{\Rcolwidth}{%
2160           \initnumbering@sectcountR%
2161           \csuse{after@pendR@the\l@dpscR}%
2162         }%
2163         \ifdefstring{\columns@position}{R}{\hfill}%
2164       }%
2165     }%
2166   }%
```

```

2166 \global\csundef{after@pendL@the\l@dpscL}%
2167 \global\csundef{after@pendR@the\l@dpscR}%
2168 }%
2169 %

```

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
`\numpagelinesR` lines on a pair of facing pages.
`\l@dminpagelines`

```

2170 \newcount\numpagelinesL
2171 \newcount\numpagelinesR
2172 \newcount\l@dminpagelines
2173
2174 %

```

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.

```

2175 \newcommand*{\Pages}[1][1,usedefault]{%
2176   \ifl@dpairing%
2177     \led@err@Pages@InsideEnv%
2178   \fi%
2179   \ifstrequal{#1}{mainmatter}{\Pages@mainmattertrue}{\Pages@mainmatterfalse}%
2180   \eledsection@correcting@skip=-2\baselineskip% line correcting for section
titles.
2181   \parledgroup@notespacing@set@correction%
2182   \typeout{}%
2183   \typeout{***** PAGES *****}%
2184   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else%
2185     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2186   \fi%
2187 %

```

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

```

2188 \cleartol@devenpage%
2189 \l@dprintingpagetrue%
2190 \begingroup%
2191 %

```


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

```
2192 \setlength{\Lcolwidth}{\textwidth}%
2193 \setlength{\Rcolwidth}{\textwidth}%
2194 %
```

```
2195 \l@dzero penalties%
2196 \endgraf\global\num@lines=\prevgraf%
2197 \global\num@linesR=\prevgraf%
2198 \global\par@line=\z@%
2199 \global\par@lineR=\z@%
2200 \global\l@dpscL=\z@%
2201 \global\l@dpscR=\z@%
2202 \writtenlinesLfalse%
2203 \writtenlinesRfalse%
2204 %
```

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

```
2205 \let\print@Xnotes\print@Xnotes@forpages%
2206 \let\print@notesX\print@notesX@forpages%
2207 %
```

Check if there are chunks to be processed.

```
2208 \check@pstarts%
2209 \loop\if@pstarts%
2210 %
```

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

```
2211 \global\advance\l@dpscL \@ne%
2212 \global\advance\l@dpscR \@ne%
2213 %
```

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

```
2214 \getlinesfromparlistL%
2215 \getlinesfromparlistR%
2216 \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2217 {\usernamecount{l@dmxlinesinpar\the\l@dpscL}}%
2218 \check@pstarts%
2219 \repeat%
2220 %
```

Zero the counts again, ready for the next bit.

```
2221 \global\l@dpscL=\z@%
2222 \global\l@dpscR=\z@%
2223 %
```

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

```

2224 \getlinesfrompagelistL%
2225 \getlinesfrompagelistR%
2226 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2227 {\l@dminpagelines}%
2228 %

```

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.

```

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

```

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

```

2232 \global\advance\l@dpscL \@ne%
2233 \global\advance\l@dpscR \@ne%
2234 \restore@pstartL@pc%
2235 \restore@pstartR@pc%
2236 %

```

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

```

2237 \global\@donereallinesL=\z@%
2238 \global\@donetotallinesL=\z@%
2239 \global\@donereallinesR=\z@%
2240 \global\@donetotallinesR=\z@%
2241 %

```

Start a loop over the boxes (chunks).

```

2242 \checkraw@text%
2243 %
2244 % \begingroup
2245 { \loop\ifaraw@text%
2246 %

```

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

```

2247 \checkpageL%
2248 \l@duselanguage{\theledlanguageL}%
2249 { \loop\ifl@dsamepage%
2250 %

```

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

```

2251 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2252 \csuse{before@pstartL@the\l@dpscL}%
2253 \global\csundef{before@pstartL@the\l@dpscL}%

```

```

2254 \do@lineL%
2255 \xifinlist{\the\l@dpscL}{\eled@sections@@}
2256 {\print@eledsectionL}%
2257 {}%
2258 \advance\numpagelinesL \@ne%
2259 %

```

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

```

2260 \ifshiftedpstarts%
2261 \ifdim\ht\l@dleftbox>0pt%
2262 \parledgroup@correction@notespacing{L}%
2263 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2264 \else%
2265 \unless\ifadvancedshiftedpstarts%
2266 \dimen0=\pagetotal%
2267 \advance\dimen0 by \baselineskip%
2268 \global\pagetotal=\dimen0%
2269 \else%
2270 \ifnomaxlines%
2271 \numdef{\@tmp}{\the\l@dpscL+1}%
2272 \ifcsdef{minpage@pstart@\@tmp}{%
2273 minpage@pstart@\@tmp}%
2274 {\dimen0=\pagetotal%
2275 \advance\dimen0 by \baselineskip%
2276 \global\pagetotal=\dimen0%
2277 }%
2278 {}%
2279 }{}%
2280 \fi%
2281 \fi%
2282 \fi%
2283 \else%
2284 \parledgroup@correction@notespacing{L}%
2285 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2286 \fi%
2287 %

```

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

```

2288 \get@nextboxL%

```

```

2289 \global\l@dskipversenumberfalse%
2290 \ifprint@last@after@pendL%
2291 \csuse{after@pendL@the\l@dpscl}%
2292 \global\csundef{after@pendL@the\l@dpscl}%
2293 \fi%
2294 \checkpageL%
2295 \checkverseL%
2296 \checkpbl%
2297 \repeat%
2298 %

```

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.

```

2299 \ifl@dpagfull%
2300 \@writelinesonpageL{\the\numpagelinesL}%
2301 \else%
2302 \@writelinesonpageL{1000}%
2303 \fi%
2304 %

```

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.

```

2305 \numpagelinesL \z@%
2306 \parledgroup@correction@notespacing@init%
2307 \clearl@dleftpage }%
2308 %

```

Now do the same for the right text.

```

2309 \checkpageR%
2310 \l@dselanguage{\theledlanguageR}%
2311 {
2312     \loop\ifl@dsamepage%
2313         \initnumbering@sectcountR%
2314         \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2315         \csuse{before@pstartR@the\l@dpscR}%
2316         \global\csundef{before@pstartR@the\l@dpscR}%
2317         \do@lineR%
2318         \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
2319         {\print@eledsectionR}%
2320         {}%
2321     \advance\numpagelinesR \@ne%
2322     \ifshiftedpstarts%
2323         \ifdim\ht\l@drightbox>0pt%
2324             \parledgroup@correction@notespacing{R}%
2325             \hb@xt@ \hsizel{\ledstrutR\unhbox\l@drightbox}%
2326         \else%
2327             \unless\ifadvancedshiftedpstarts%

```

```

2327         \dimen0=\pagetotal%
2328         \advance\dimen0 by \baselineskip%
2329         \global\pagetotal=\dimen0%
2330     \else%
2331         \ifnomaxlines%
2332             \numdef{\@tmp}{\the\l@dpscR+1}%
2333             \ifcsdef{minpage@pstart@\@tmp}{%
2334                 \ifnumless{\the\c@page}{\csuse{
minpage@pstart@\@tmp}}}%
2335                 {\dimen0=\pagetotal%
2336                 \advance\dimen0 by \baselineskip%
2337                 \global\pagetotal=\dimen0%
2338                 }%
2339                 {%
2340                 }%
2341             \fi%
2342         \fi%
2343     \fi%
2344 \else%
2345     \parledgroup@correction@notespacing{R}%
2346     \hb@xt@ \hsizel{\ledstrutR\unhbox\l@drightrightbox}%
2347 \fi%
2348 \get@nextboxR%
2349 \global\l@dskipversenumberRfalse%
2350 \ifprint@last@after@pendR%
2351     \csuse{after@pendR@\the\l@dpscR}%
2352     \global\csundef{after@pendR@\the\l@dpscR}%
2353 \fi%
2354 \checkpageR%
2355 \checkverseR%
2356 \checkpbR%
2357 \repeat%
2358 \ifl@dpagfull%
2359     \@writelinesonpageR{\the\numpagelinesR}%
2360 \else%
2361     \@writelinesonpageR{1000}%
2362 \fi%
2363 \numpagelinesR=\z@%
2364 \parledgroup@correction@notespacing@init%
2365 %

```

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

```

2366     \clearl@drightrightpage}%
2367 %

```

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.

```

2368     \checkraw@text%
2369     \ifaraw@text%
2370         \getlinesfrompagelistL%

```

```

2371 \getlinesfrompagelistR%
2372 \l@dcalcm@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2373 \l@dminpagelines}%
2374 \fi%
2375 \repeat}%
2376 %

```

We have now output the text from all the chunks.

```

2377 \fi%
2378 %

```

Make sure that there are no inserts hanging around.

```

2379 \flush@notes%
2380 \flush@notesR%
2381 \endgroup%
2382 %

```

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

```

2383 \global\l@dpscL=\z%
2384 \global\l@dpscR=\z%
2385 \global\l@dnumstartsL=\z%
2386 \global\l@dnumstartsR=\z%
2387 \global\instanzaLfalse%
2388 \global\instanzaRfalse%
2389 \l@dprintingpagesfalse%
2390 \finish@Pages@notes%Needed to prevent final notes overlap line number
2391 \ignorespaces}
2392
2393
2394 %

```

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.

```

2395 \newcommand{\finish@Pages@notes}{%
2396 \def\do##1{%
2397 %

```

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

```

2398 \ifnocritical@%
2399 \global\newnamebox{##1footins}
2400 \fi
2401 \ifnofamiliar@%
2402 \global\newnamebox{footins##1}
2403 \fi
2404 %

```

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

```

2405 \ifvoid\csuse{##1footins}%
2406 \ifvoid\csuse{footins##1}\else%
2407 \newpage\null%
2408 \listbreak%
2409 \fi%
2410 \else%
2411 \newpage\null%
2412 \listbreak%
2413 \fi%
2414 }%
2415 \dolistloop{\@series}%
2416 }%
2417 %

```

XIX.4 Struts

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

```

\ledstrutR
2418 \newcommand*{\ledstrutL}{%
2419 \newcommand*{\ledstrutR}{%
2420
2421 %

```

XIX.5 Page clearing

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

```

2422 \providecommand{\cleartoevenpage}[1][\@empty]{%
2423 \clearpage
2424 \ifodd\c@page\hbox{##1}\clearpage\fi}
2425
2426 \newcommand*{\cleartol@evenpage}{%
2427 \ifdim\pagetotal<\topskip% on an empty page
2428 \else
2429 \clearpage
2430 \Pages@mainmatter%
2431 \fi
2432 \ifodd\c@page%
2433 \ifprevpgnotnumbered%
2434 \addtocounter{par@page}{-1}%
2435 \ifdef\prevpgstyle{\thispagestyle{\prevpgstyle}}}%
2436 \fi%
2437 \hbox{##1}\clearpage%
2438 \fi%
2439 }%
2440 %

```

`\clearl@leftpage` and `\clearl@rightpage` 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`

```

2441 \newcommand*\clearl@leftpage{%
2442   \ifdim\pagetotal=0pt\hbox{}\fi%
2443   \newpage%
2444   \ifodd\c@page\else
2445     \led@err@LeftOnRightPage
2446     \hbox{}%
2447     \cleardoublepage
2448   \fi}
2449
2450 \newcommand*\clearl@rightpage{%
2451   \ifdim\pagetotal=0pt\hbox{}\fi%
2452   \newpage%
2453   \ifodd\c@page
2454     \led@err@RightOnLeftPage
2455     \hbox{}%
2456     \cleartoevenpage
2457   \fi}
2458
2459 %

```

XIX.6 Lines managing

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

```

\@cs@linesinparL
\@cs@linesinparR
2460 \newcommand*\getlinesfromparlistL{%
2461   \ifx\linesinpar@listL\empty
2462     \gdef\@cs@linesinparL{0}%
2463   \else
2464     \gl@p\linesinpar@listL\to\@cs@linesinparL
2465   \fi}
2466 \newcommand*\getlinesfromparlistR{%
2467   \ifx\linesinpar@listR\empty
2468     \gdef\@cs@linesinparR{0}%
2469   \else
2470     \gl@p\linesinpar@listR\to\@cs@linesinparR
2471   \fi}
2472
2473 %

```

`\getlinesfrompagelistL` and `\getlinesfrompagelistR` get the next entry from the `\linesonpage@listL` and `\linesonpage@listR` and puts it into `\@cs@linesonpageL` and `\@cs@linesonpageR`; if the list is empty, it sets `\@cs@linesonpageL` and `\@cs@linesonpageR` to 0. Similarly for `\getlinesfrompagelistR`.

puts it into `\@cs@linesonpageL`; if the list is empty, it sets `\@cs@linesonpageL` to 1000. Similarly for `\getlinesfrompagelistR`.

```

2474 \newcommand*\getlinesfrompagelistL{%
2475   \ifx\linesonpage@listL\empty
2476     \gdef\@cs@linesonpageL{1000}%
2477   \else
2478     \glp\linesonpage@listL\to\@cs@linesonpageL
2479   \fi}
2480 \newcommand*\getlinesfrompagelistR{%
2481   \ifx\linesonpage@listR\empty
2482     \gdef\@cs@linesonpageR{1000}%
2483   \else
2484     \glp\linesonpage@listR\to\@cs@linesonpageR
2485   \fi}
2486
2487 %

```

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

```

2488 \newcommand*\@writelinesonpageL}[1]{%
2489   \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2490   \next}
2491 \newcommand*\@writelinesonpageR}[1]{%
2492   \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2493   \next}
2494
2495 %

```

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

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

```

2496 \newcommand*\l@dcalc@maxoftwo}[3]{%
2497   \ifnum #2>#1\relax
2498     #3=#2\relax
2499   \else
2500     #3=#1\relax
2501   \fi}
2502 \newcommand*\l@dcalc@minoftwo}[3]{%
2503   \ifnum #2<#1\relax
2504     #3=#2\relax
2505   \else
2506     #3=#1\relax
2507   \fi}
2508
2509 %

```

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@dpagefull` is set FALSE and `\l@dsamepagetrue` `\l@dsamepagefalse` `\ifl@dsamepage` is set TRUE. If the page is spatially full then `\ifl@dpagefull` 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@dpagefull` and `\ifl@dsamepage` are set FALSE.

```

\checkpageL
\checkpageR
2510 \newif\ifl@dsamepage
2511 \l@dsamepagetrue
2512 \newif\ifl@dpagefull
2513
2514 \newcommand*{\checkpageL}{%
2515   \l@dpagefulltrue
2516   \l@dsamepagetrue
2517   \check@goal
2518   \ifdim\pagetotal<\ledthegoal
2519     \ifnum\numpagelinesL<\l@dsamepagelines
2520     \else
2521       \ifnomaxlines%
2522       \else%
2523         \l@dsamepagefalse%
2524         \l@dpagefullfalse%
2525       \fi%
2526     \fi
2527   \else
2528     \l@dsamepagefalse
2529     \l@dpagefulltrue
2530   \fi%
2531   \ifprint@last@after@pendL%
2532     \l@dpagefullfalse%
2533     \l@dsamepagefalse%
2534     \print@last@after@pendLfalse%
2535   \fi%
2536 }%
2537
2538 \newcommand*{\checkpageR}{%
2539   \l@dpagefulltrue
2540   \l@dsamepagetrue
2541   \check@goal
2542   \ifdim\pagetotal<\ledthegoal
2543     \ifnum\numpagelinesR<\l@dsamepagelines
2544     \else
2545       \ifnomaxlines%
2546       \else%
2547         \l@dsamepagefalse%
2548         \l@dpagefullfalse%
2549       \fi%
2550     \fi

```

```

2551 \else
2552 \l@dsamepagefalse
2553 \l@dpagetrue
2554 \fi%
2555 \ifprint@last@after@pendR%
2556 \l@dpagetrue%
2557 \l@dsamepagefalse%
2558 \print@last@after@pendRfalse%
2559 \fi%
2560 }%
2561
2562 %

```

\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.

```

2563 \newcommand{\checkpbL}{
2564 \IfStrEq{\ledpb@setting}{after}{
2565 \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@dpagetrue\
l@dsamepagefalse}{
2566 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@dpagetrue\
l@dsamepagetrue}{
2567 }{
2568 \IfStrEq{\ledpb@setting}{before}{
2569 \numdef{\next@absline}{\the\absline@num+1}
2570 \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagetrue\
l@dsamepagefalse}{
2571 \xifinlistcs{\next@absline}{l@prev@nopb}{\l@dpagetrue\
l@dsamepagetrue}{
2572 }{
2573 }
2574 }
2575 \newcommand{\checkpbR}{
2576 \IfStrEq{\ledpb@setting}{after}{
2577 \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@dpagetrue\
l@dsamepagefalse}{
2578 \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@dpagetrue\
l@dsamepagetrue}{
2579 }{
2580 \IfStrEq{\ledpb@setting}{before}{
2581 \numdef{\next@abslineR}{\the\absline@numR+1}
2582 \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@dpagetrue\
l@dsamepagefalse}{
2583 \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@dpagetrue\
l@dsamepagetrue}{
2584 }{
2585 }
2586 %

```

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

```

2587 \newcommand{\checkverseL}{
2588 \ifinstanzaL
2589   \iflednopbinverse
2590     \ifinserthangingsymbol
2591       \numgdef{\prev@abslineverse}{\the\absline@num-1}
2592       \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{}
2593       \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\prev@abslineverse}\fi}{}
2594     \fi
2595   \fi
2596 \fi
2597 }
2598 \newcommand{\checkverseR}{
2599 \ifinstanzaR
2600   \iflednopbinverse
2601     \ifinserthangingsymbolR
2602       \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2603       \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{}
2604       \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\prev@abslineverse}\fi}{}
2605     \fi
2606   \fi
2607 \fi
2608 }
2609 %

```

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

```

\check@goal
2610 \newdimen\ledthegoal
2611 \ifshiftedpstarts
2612   \newcommand*{\@goalfraction}{0.95}
2613 \else
2614   \newcommand*{\@goalfraction}{0.9}
2615 \fi
2616
2617 \newcommand*{\check@goal}{%
2618   \ledthegoal=\@goalfraction\pagegoal}
2619 \newcommand{\setgoalfraction}[1]{%
2620   \xdef\@goalfraction{#1}%
2621 }
2622 %

```

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

```

\ifwrittenlinesL
2623 \newif\ifwrittenlinesL

```

```

2624 \newif\ifwrittenlinesR
2625
2626 %

```

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.

```

2627 \newif\if@getnextbox%
2628 %

```

`\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.

```

2629 \newcommand*{\get@nextboxL}{%
2630   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}% box is not empty
2631 %

```

The current box is not empty; do nothing.

```

2632   \else%                               box is empty
2633 %

```

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

```

2634   \@getnextboxtrue%Should be local, but be cautious
2635 %

```

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

```

2636   \ifnum\usenamecount{\l@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesL
2637   \parledgroup@notes@endL%
2638   \unless\ifnosyncpstarts%
2639   \@getnextboxfalse%
2640 %

```

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.

```

2641   \ifnomaxlines%
2642   \ifdim\pagetotal<\ledthegoal%
2643   \numdef{\@tmp}{\l@dpscl+1}%
2644   \ifcsdef{afterlines@pstart@\@tmp R}{%

```

```

2645 \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp R}}
2646 %
2647 {}%
2648 {\ifcsdef{minpage@pstart@\@tmp}%
2649   {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2650     {\ifnum\numpagelinesL=\l@dminpagelines%
2651       \@getnextboxtrue%
2652     }%
2653     {\@getnextboxtrue}}%
2654     {\@getnextboxtrue}}%
2655   }%
2656 }%
2657 {}%
2658 \fi%
2659 \fi%
2660 \fi%
2661 \else%
2662   \ifnomaxlines%
2663     \numdef{\@tmp}{\the\l@dpscl+1}%
2664     \ifcsdef{minpage@pstart@\@tmp}%
2665       \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2666         {\ifdimgreater{\pagetotal}{\ledthegoal}%
2667           {\@getnextboxtrue}%
2668           {\@getnextboxfalse}%
2669         }%
2670         {\@getnextboxtrue}%
2671       }{}%
2672     \fi%
2673   \fi%
2674 %

```

Sufficient lines have been output.

```

2675 \if@getnextbox%
2676   \ifnum\usernamecount\l@dmaxlinesinpar\the\l@dpscl=\@donetotallinesL
2677     \parledgroup@notes@endL
2678   \fi
2679   \ifwrittenlinesL\else
2680 %

```

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

```

2681   \@writelinesinparL
2682   \writtenlinesLtrue
2683 \fi
2684 \ifnum\l@dnumstartsL>\l@dpscl
2685 %

```

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

```

2686     \writtenlinesLfalse
2687     \ifbypstart@
2688         \global\line@num=0%
2689         \resetprevline@%
2690     \fi
2691 % Add the content of the optional argument of the previous \protect\cs{pend
2692 }.
2693 % \begin{macrocode}
2694     \csuse{after@pendL@the\l@dpscl}%
2695     \global\csundef{after@pendL@the\l@dpscl}%
2696 %

```

Check the number of lines

```

2696     \l@dcalcm@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2697         {\the\@donetotallinesL}%
2698         {\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2699     \global\@donetotallinesL \z@
2700 %

```

Go to the next pstart

```

2701     \global\advance\l@dpscl \@ne
2702     \global\pstartnumtrue%
2703     \restore@pstartL@pc%
2704 %

```

Add notes of parallel ledgroup.

```

2705     \parledgroup@notes@endL
2706     \parledgroup@correction@notes@spacing@final{L}
2707     \else
2708 %

```

```

2709     \print@last@after@pendLtrue%
2710 \fi
2711 \fi
2712 \fi}
2713 %

```

```

2714 \newcommand*{\get@nextboxR}{%
2715 \ifvbox\namebox{l@dRcolrawbox\the\l@dpscl}% box is not empty
2716 \else% box is empty
2717 \getnextboxtrue%
2718 \ifnum\usernamecount{l@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesR
2719 \parledgroup@notes@endR
2720 \unless\ifnosyncpstarts%
2721 \getnextboxfalse%
2722 \ifnomaxlines%
2723 \ifdim\pagetotal<\ledthegoal%
2724 \numdef{\@tmp}{\l@dpscl+1}%
2725 \ifcsdef{afterlines@pstart@\@tmp L}{%

```

```

2726 \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp L}}
2727 %
2728 {}%
2729 {\ifcsdef{minpage@pstart@\@tmp}%
2730 {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2731 {\ifnum\numpagelinesR=\l@dminpagelines%
2732 \getnextboxtrue%
2733 \fi%
2734 }%
2735 {\getnextboxtrue}}%
2736 }%}%
2737 }%
2738 {}%
2739 \fi%
2740 \fi%
2741 \fi%
2742 \else%
2743 \ifnomaxlines%
2744 \numdef{\@tmp}{\the\l@dpscR+1}%
2745 \ifcsdef{minpage@pstart@\@tmp}%
2746 \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2747 {\ifdimgreater{\pagetotal}{\ledthegoal}%
2748 {\getnextboxtrue}%
2749 {\getnextboxfalse}%
2750 }%
2751 {\getnextboxtrue}%
2752 }{}
2753 \fi%
2754 \fi%
2755 \if@getnextbox%
2756 \ifnum\usenamecount{l@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2757 \parledgroup@notes@endR
2758 \fi
2759 \ifwrittenlinesR\else
2760 \writelinesinparR
2761 \writtenlinesRtrue
2762 \fi
2763 \ifnum\l@dnumpstartsR>\l@dpscR
2764 \writtenlinesRfalse
2765 \ifbypstart@R
2766 \global\line@numR=0%
2767 \resetprevline@%
2768 \fi
2769 \csuse{after@pendR@\the\l@dpscR}%
2770 \global\csundef{after@pendR@\the\l@dpscR}%
2771 \l@dcalc@maxoftwo{\the\usenamecount{l@dmaxlinesinpar\the\l@dpscR}}%
2772 {\the\@donetotallinesR}%
2773 {\usenamecount{l@dmaxlinesinpar\the\l@dpscR}}%
2774 \global\@donetotallinesR \z@

```



```

2775 \global\advance\l@dpscr \@ne
2776 \global\pstartnumRtrue%
2777 \restore@pstartR@pc%
2778 \parledgroup@notes@endR
2779 \parledgroup@correction@notes@spacing@final{R}
2780 \else
2781 \print@last@after@pendRtrue%
2782 \fi
2783 \fi
2784 \fi}
2785
2786 %

```

XX Page numbering

XX.1 Global options

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

```

2787
2788 \newcommand{\par@patch@thepage}{%
2789 \ifbool{expr}%
2790   bool{sameparallelpagnumber}%
2791   or bool{prevpgnotnumbered}%
2792 }%
2793 {%
2794 \patchcmd{\thepage}%
2795   {page}{par@page}%

```

```

2796     {}%
2797     {\led@error@fail@patch@thepage}%
2798   }{}%
2799 }%
2800
2801 \newcommand{\par@patch@pagenumbering}{%
2802   \ifboolexpr{%
2803     bool{sameparallelnumber}%
2804     or bool{prevpgnotnumbered}%
2805   }%
2806   {%
2807     \setcounter{par@page}{1}%
2808   }%
2809   {}%
2810   \par@patch@thepage%
2811 }%
2812
2813 \ifl@dmemoir%
2814   \apptocmd{\@mempnum}%
2815     {\par@patch@pagenumbering}%
2816     {}%
2817     {\led@error@fail@patch@@mempnum}%
2818
2819 \else%
2820   \apptocmd{\pagenumbering}%
2821     {\par@patch@pagenumbering}%
2822     {}%
2823     {\led@error@fail@patch@pagenumbering}%
2824 \fi%
2825
2826 \AtBeginDocument{\par@patch@thepage}%
2827 %

```

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

```

2828 \AtBeginDocument{%
2829   \apptocmd{\@outputpage}{%
2830     \ifsameparallelnumber%
2831       \ifl@dprintingpages%
2832         \ifodd\c@page\else%
2833           \stepcounter{par@page}%
2834         \fi%
2835       \else%
2836         \stepcounter{par@page}%
2837       \fi%
2838     \else%
2839       \stepcounter{par@page}%
2840     \fi%

```

```

2841 }%
2842 {}%
2843 {\led@error@fail@patch@@@outputpage}%
2844 }
2845 %

```

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

```

2846 \newcounter{par@page}%
2847 \setcounter{par@page}{1}%
2848 %

```

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@devenpage.

```

\ifPages@mainmatter \newif\ifPages@mainmatter
\Pages@mainmatter \newcommand{\Pages@mainmatter}{%
2851   \ifPages@mainmatter%
2852   \pagenumbering{arabic}%
2853   \addtocounter{page}{1}%
2854   \addtocounter{par@page}{-1}%
2855   \patchcmd{\thepage}{page}{par@page}{}{}%
2856   \fi%
2857 }
2858 %

```

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 \pretocmd{\chapter}{%
2860   \ifl@dprintingpages%
2861   \vbox{}%
2862   \fi%
2863 }%
2864 {}%
2865 {}%
2866 %

```

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

```

2867 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2868 \eledsectnotoc{R}
2869 %

```

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

```

2870 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2871 \eledsectmark{L}
2872 %

```

\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.

```

2873 \newskip\eledsection@correcting@skip
2874 %

```

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

```

2875 \newwrite\eled@sectioningR@out
2876 %

```

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).

```

2877 \def\l@prev@pbR{}
2878 \def\l@prev@nopbR{}
2879 %

```

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

```

2880 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2881 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2882 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2883 \newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\led@nopbnumR{#1}}}
2884 %

```

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

```
2885 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2886 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{\#1}}
2887 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2888 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{\#1}}
2889 %
```

XXIII Parallel ledgroup

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

```
2890 \newmarks\parledgroup@
2891 \newmarks\parledgroup@series
2892 \newmarks\parledgroup@type
2893 %
```

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

```
2894 \newcommand{\parledgroup@notes@startL}{%
2895   \ifnum\usernamecount{1@dmxlinesinpar\the\l@dpscL}>0%
2896     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{%
2897     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{%
2898     \fi%
2899     \global\ledgroupnotesL@true%
2900     \insert@noterule@ledgroup{L}%
2901 }
2902 \newcommand{\parledgroup@notes@startR}{%
2903   \ifnum\usernamecount{1@dmxlinesinpar\the\l@dpscR}>0%
2904     \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{%
2905     \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{%
2906     \fi%
2907     \global\ledgroupnotesR@true%
2908     \insert@noterule@ledgroup{R}%
2909 }
2910 %
```

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

```

2911 \newcommand{\parledgroup@notes@endL}{%
2912   \global\ledgroupnotesL@false%
2913 }
2914 \newcommand{\parledgroup@notes@endR}{%
2915   \global\ledgroupnotesR@false%
2916 }
2917 %

```

`\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`

```

2918 \newcommand{\insert@noterule@ledgroup}[1]{
2919   \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2920     \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2921       \csuse{ifledgroupnotes#1@}
2922       \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2923       \csuse{\splitbotmarks\parledgroup@series footnoterule}
2924       \fi
2925     }
2926     {}
2927     \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2928       \csuse{ifledgroupnotes#1@}
2929       \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2930       \csuse{footnoterule\splitbotmarks\parledgroup@series}
2931       \fi
2932     }{}
2933   }
2934   {}
2935 }
2936 %

```

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

```

2937 \newcommand{\setparledgroupnotespacing}[1]{\gdef\@parledgroupnotespacing
2938   {#1}}
2939 \newcommand{\@parledgroupnotespacing}{}
2940 %

```

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

```

2940 \dimdef{\parledgroup@notespacing@correction}{0pt}
2941 \newcommand{\parledgroup@notespacing@set@correction}{%
2942   {\@getfirstseries\csuse{Xnotefontsize@\@firstseries}}%We suppose all the
series has the same footnote size setup

```

```

2943 \parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
2944 \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing}
2945 }%
2946 %

```

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

```

2947 \newcommand{\parledgroup@correction@notespacing@init}{
2948   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2949   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2950 }
2951 \parledgroup@correction@notespacing@init
2952 %

```

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

```

2953 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2954   \ifparledgroup
2955     \vspace{\parledgroup@notespacing@correction@accumulated}
2956     \parledgroup@correction@notespacing@init%
2957     \ifstrequal{#1}{L}{
2958       \numdef{\@checking}{\the\l@dpscl-1}
2959     }{
2960       \numdef{\@checking}{\the\l@dpscr-1}
2961     }
2962     \dimdef{\@beforenotes@current@diff}{\csuse{\parledgroup@beforenotes@
2963       \@checking L}-\csuse{\parledgroup@beforenotes@
2964       \@checking R}}%
2965     \ifstrequal{#1}{L}{
2966       {% Left
2967         \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\
2968           \@beforenotes@current@diff}}%
2969       }%
2970       {% Right
2971         \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\
2972           \@beforenotes@current@diff}}{}
2973       }%
2974     }
2975   }
2976 }

```

\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@correction

and `\parledgroup@notespacing@correction@modulo`. If `\parledgroup@notespacing@correction@modulo` is equal or greater than `\baselineskip`:

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

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

```

2973 \newcommand{\parledgroup@correction@notespacing}[1]{%
2974   \csuse{ifledgroupnotes#1@}%
2975   \vspace{-\parledgroup@notespacing@correction}%
2976   \dimdef{\parledgroup@notespacing@correction@accumulated}{\
parledgroup@notespacing@correction@accumulated+\
parledgroup@notespacing@correction}%
2977   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo+\
parledgroup@notespacing@correction}%
2978   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip
}{\advance\numpagelinesL -\@ne%
2979   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo-\baselineskip}%
2980   }% mean greater than equal
2981   \fi%
2982 }
2983 %

```

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

```

2984 \dimdef\parledgroup@beforenotesL{0pt}
2985 \dimdef\parledgroup@beforenotesR{0pt}
2986 %

```

`\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.

```

2987 \newcommand{\parledgroup@beforenotes@save}[1]{
2988   \ifparledgroup
2989     \csdimgdef{@parledgroup@beforenotes@the\csuse{ldnumpstarts#1}#1}{\
csuse{\parledgroup@beforenotes#1}}
2990     \csdimgdef{\parledgroup@beforenotes#1}{0pt}
2991   \fi
2992 }
2993 %

```


XXIV Compatibility with eledmac

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

```

2994 \ifeledmaccompat@%
2995
2996
2997 \unless\ifnocritical@
2998 \let\onlyXside\Xonlyside
2999 \fi
3000 \fi
3001 %

```

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.

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	105
COMMAND\@Rlineflag	65, 70
COMMAND\@adv	35, 138
COMMAND\@cs@linesinparL	96
COMMAND\@cs@linesonpageL	97
COMMAND\@eledsectionL	80
COMMAND\@eledsectionR	80
COMMAND\@eledsectmark	108
COMMAND\@eledsectnotoc	107
COMMAND\@footnotetext	72

COMMAND\@goalfraction	12, 100
COMMAND\@l@tempcnta	64
COMMAND\@lab	33, 70, 138
COMMAND\@lopL	39, 91, 97
COMMAND\@lopR	39, 97
COMMAND\@mempnum	105
COMMAND\@namedef	74
COMMAND\@namuse	74
COMMAND\@nl	34, 40, 70, 138
COMMAND\@nl@regR	34
COMMAND\@outputpage	105, 106
COMMAND\@page	70
COMMAND\@par@sync@option	33
COMMAND\@parledgroupnotespacing	110
COMMAND\@pend	38
COMMAND\@pendR	38
COMMAND\@pstart	38
COMMAND\@pstartstrue	78
COMMAND\@ref	36–38, 40, 138
COMMAND\@ref@regR	37
COMMAND\@set	35, 138
COMMAND\@sw	37
COMMAND\AtBeginPairs	8, 45, 136
COMMAND\AtEveryPend	136–138
COMMAND\AtEveryPstart	2, 16, 52, 81, 136–138
COMMAND\AtEveryPstartCall	2, 16, 52, 137
COMMAND\Clear the right lines for \read@linelist	33
COMMAND\Columns	7, 8, 16, 44, 80, 84, 133–136, 138, 139
COMMAND\Columns@print@after@pend	87
COMMAND\Columns@print@before@pstart	87
COMMAND\Lcolwidth	8, 10, 89
COMMAND\Leftsidehook	133
COMMAND\Leftsidehookend	133
COMMAND\Pages	4, 7, 9, 10, 12, 16, 44, 61, 65, 67, 88, 89, 94, 105, 107, 110, 133, 136–139
COMMAND\Pages@mainmatter	107
COMMAND\Rcolwidth	8, 10, 89
COMMAND\Rightsidehook	133
COMMAND\Rightsidehookend	133
COMMAND\Rlineflag	114
COMMAND\Xmaxhnotes	13
COMMAND\Xnoteswidthliketwocolumns	9, 136
COMMAND\Xonlyside	13, 65, 115
COMMAND\&	17
COMMAND\absline@numR	32
COMMAND\add@penalties	64
COMMAND\add@penaltiesL	63
COMMAND\advanceline	35, 41, 138
COMMAND\affixline@num	59
COMMAND\affixline@numR	59, 133, 134
COMMAND\affixpstart@num	61

COMMAND\affixstart@numR	61
COMMAND\affixside@note	71
COMMAND\aftercolumnseparator	9, 84, 135
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, 138
COMMAND\beforecolumnseparator	9, 84, 135
COMMAND\begin	17
COMMAND\beginnumbering	14–16, 25, 33, 84, 134, 135, 138
COMMAND\beginnumberingR	40
COMMAND\bf	134
COMMAND\bfseries	134
COMMAND\brokenpenalty	64
COMMAND\chapter	107, 133
COMMAND\check@goal	100
COMMAND\check@pstarts	78
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, 137
COMMAND\cleartoevenpage	95
COMMAND\cleartol@devenpage	95, 107
COMMAND\columnrulewidth	8, 84
COMMAND\columns@position	84
COMMAND\columnseparator	8
COMMAND\columnspan	9, 135
COMMAND\correct@Xfootins@box	67, 68, 137
COMMAND\correct@footinsX@box	67, 137
COMMAND\critext	137
COMMAND\csname	44
COMMAND\displaywidowpenalty	64
COMMAND\do@actions	57
COMMAND\do@actions@fixedcode	133
COMMAND\do@actions@nextR	57
COMMAND\do@actionsR	57, 133
COMMAND\do@ballast	64
COMMAND\do@ballastR	57
COMMAND\do@insidelineLhook	135
COMMAND\do@insidelineRhook	135

COMMAND\do@line	52
COMMAND\do@line(L/R)	55
COMMAND\do@lineL	53, 63, 133, 134
COMMAND\do@lineLhook	133
COMMAND\do@lineR	55, 133–135
COMMAND\do@lineRhook	133
COMMAND\do@lockoff	139
COMMAND\do@lockoffR	36
COMMAND\do@lockon	138
COMMAND\do@lockonR	35
COMMAND\doinsidelineLhook	136
COMMAND\doinsidelineRhook	136
COMMAND\dolineLhook	136
COMMAND\dolineRhook	136
COMMAND\edindex	137
COMMAND\edlabel	134, 137
COMMAND\edtext	36, 37, 40, 41, 136, 137
COMMAND\eled@sectioningR@out	108
COMMAND\eledchapter	137
COMMAND\eledsection	136, 137, 139
COMMAND\eledsection@correcting@skip	108
COMMAND\eledsectmark	19, 108
COMMAND\eledsectnotoc	19, 107
COMMAND\eledxxx	136
COMMAND\end	17
COMMAND\endgraf	51
COMMAND\endlock	41, 138
COMMAND\endnumbering	14, 16, 26, 138
COMMAND\endsub	40, 138
COMMAND\endumbering	14
COMMAND\expandafter	42
COMMAND\extensionchars	24
COMMAND\firstlinenum	15, 135, 139
COMMAND\firstsublinenum	15, 135, 139
COMMAND\fix@page	35, 138
COMMAND\flag@end	40, 133, 136
COMMAND\flag@start	40, 136
COMMAND\flush@notesR	64
COMMAND\footnoteX	42
COMMAND\footnoteXmk	13
COMMAND\footnoteXnomk	13, 42
COMMAND\frontmatter	12, 19
COMMAND\get@nextboxL	134
COMMAND\get@nextboxR	134
COMMAND\getline@numL	56
COMMAND\getline@numR	56
COMMAND\getlinesfrompagelistL	96
COMMAND\getlinesfrompagelistR	97
COMMAND\getlinesfromparlistL	96
COMMAND\getlinesfromparlistR	96

COMMAND\gl@p	42
COMMAND\goalfraction	114
COMMAND\hangingsymbol	114, 134
COMMAND\hfill	83, 84
COMMAND\hidenumbering	15, 138
COMMAND\if@getnextbox	101
COMMAND\ifPages@mainmatter	107
COMMAND\ifbypage@	138
COMMAND\ifbypstart@R	138
COMMAND\ifdim	83
COMMAND\ifinserthangingsymbol	72
COMMAND\ifinserthangingsymbolR	72
COMMAND\ifl@dpagefull	98
COMMAND\ifl@dpageing	22, 136
COMMAND\ifl@dpairing	22, 133
COMMAND\ifl@dsamelang	135
COMMAND\ifl@dsamepage	98
COMMAND\ifl@pagefull	91
COMMAND\ifledRcol	22
COMMAND\iflledRcol	134
COMMAND\ifnumberedpar@	47
COMMAND\ifnumberingR	134
COMMAND\ifnumberpstart	44
COMMAND\ifpst@rtedL	25, 26, 48, 133
COMMAND\ifpst@rtedR	25
COMMAND\ifsublines@	35
COMMAND\insert@countR	36
COMMAND\insert@noterule@ledgroup	110
COMMAND\insertlines@list	36
COMMAND\insertlines@listR	37
COMMAND\inserts@list	48
COMMAND\inserts@listR	63
COMMAND\l@d@nums	41, 65
COMMAND\l@d@set	35, 41, 138
COMMAND\l@dLcolrawbox	47
COMMAND\l@dLcolrawbox1	77
COMMAND\l@dLcolrawbox2	77
COMMAND\l@dRcolrawbox	47
COMMAND\l@dbfnote	72, 138
COMMAND\l@dcalc@maxoftwo	97
COMMAND\l@dcalc@minoftwo	97
COMMAND\l@dchecklang	133, 136
COMMAND\l@dcsnote	135
COMMAND\l@dleftbox	52, 91, 137
COMMAND\l@dlinenumR	31, 133
COMMAND\l@dlsnote	135
COMMAND\l@dmake@labels	70
COMMAND\l@dmaxlinesinpar	89
COMMAND\l@dmaxlinesinpar1	77
COMMAND\l@dminpagelines	90, 133

COMMAND\l@dnumstartsl	77, 133
COMMAND\l@dprintingcolumnstrue	137
COMMAND\l@dprintingpagestrue	137
COMMAND\l@dpscL	81, 89, 90, 102
COMMAND\l@dpscR	81, 89, 90
COMMAND\l@drsnote	135
COMMAND\l@dsetupmaxlinecounts	77
COMMAND\l@duselanguage	75, 76, 133
COMMAND\l@dzeromaxlinecounts	77
COMMAND\l@prev@nopbR	108
COMMAND\l@prev@pbR	108
COMMAND\labelstarttrue	134
COMMAND\labelref@list	70
COMMAND\labelref@listR	70
COMMAND\lang	75
COMMAND\last@page@numR	35
COMMAND\led	134
COMMAND\led@nopbR	108, 109
COMMAND\led@nopbnumR	108, 109
COMMAND\led@pbR	108, 109
COMMAND\led@pbnumR	108, 109
COMMAND\ledinnerrote	18
COMMAND\ledleftnote	18
COMMAND\lednopb	17, 83, 99
COMMAND\lednopbR	108
COMMAND\lednopbnumR	108
COMMAND\ledouterote	18
COMMAND\ledpb	83, 99
COMMAND\ledpbR	108
COMMAND\ledpbnumR	108
COMMAND\ledrightnote	18
COMMAND\ledsidenote	18
COMMAND\ledstrutL	133
COMMAND\ledstrutR	133, 139
COMMAND\ledthegoal	100
COMMAND\ledtrutL	133, 139
COMMAND\leftlinenumR	31, 133
COMMAND\let	42
COMMAND\line@list@R	37
COMMAND\line@list@stuff	33, 40
COMMAND\line@margin	29
COMMAND\line@marginR	29, 133
COMMAND\line@numR	32
COMMAND\lineation	15, 137
COMMAND\lineation*	15, 28, 136
COMMAND\lineationR	15, 28, 137
COMMAND\linenum@out	70
COMMAND\linenum@outR	39
COMMAND\linenumberstyle	15, 115
COMMAND\linenumberstyle*	115

COMMAND\linenumberstyleR	15, 115
COMMAND\linenumincrement	15, 135, 139
COMMAND\linenummargin	15, 29, 133, 138
COMMAND\linenumrepR	31, 133
COMMAND\linesinpar@listL	38, 96
COMMAND\linesonpage@listL	39, 96
COMMAND\lock@off	36
COMMAND\lock@on	35
COMMAND\mainmatter	2, 12, 19, 139
COMMAND\makeatletter	55
COMMAND\maxchunks	7, 17, 77, 78
COMMAND\maxhnotesX	13
COMMAND\memorydump	14, 27
COMMAND\n@num	137
COMMAND\new@lineL	40
COMMAND\new@lineR	40
COMMAND\newhookcommand@series	43
COMMAND\newif	137
COMMAND\newpage	95, 96, 137
COMMAND\newseries	44
COMMAND\newseries@	41
COMMAND\newseries@par	41, 43, 44
COMMAND\noeledxxx	136
COMMAND\nomark@	42
COMMAND\nomaxlines	38
COMMAND\normalbfnoteX	133, 138
COMMAND\notesXwidthliketwocolumns	9, 136
COMMAND\num@lines	64
COMMAND\num@lines(R)	47
COMMAND\numberingR	26
COMMAND\numberlinefalse	7
COMMAND\numberonlyfirstinline	134
COMMAND\numberpstartfalse	15
COMMAND\numberpstarttrue	15, 134, 139
COMMAND\one@line	47, 72
COMMAND\one@lineR	47
COMMAND\onlyXside	115
COMMAND\onlysideX	13, 65, 138
COMMAND\otherlanguage	139
COMMAND\page@action	35, 138
COMMAND\pagenumbering	105, 139
COMMAND\pages	12
COMMAND\pagetotal	91, 137
COMMAND\par@line	64
COMMAND\par@line(R)	47
COMMAND\par@patch@pagenumbering	105
COMMAND\par@patch@thepage	105
COMMAND\par@sync@option	21
COMMAND\parledgroup@	109
COMMAND\parledgroup@beforenotes@save	112

COMMAND\parledgroup@beforenotesL	112
COMMAND\parledgroup@beforenotesR	112
COMMAND\parledgroup@correction@notespacing	111
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	110, 111
COMMAND\parledgroup@notespacing@correction@accumulated	111
COMMAND\parledgroup@notespacing@correction@modulo	112
COMMAND\parledgroup@notespacing@set@correction	110
COMMAND\parledgroup@series	109
COMMAND\parledgroup@type	109
COMMAND\parledgroupnotespacing	114
COMMAND\parledgrouptrue	18
COMMAND\patchcmd	138
COMMAND\pausenumbering	27
COMMAND\pend	3, 7, 10, 16–18, 44, 47, 48, 50–52, 77, 87, 135, 136, 138, 139
COMMAND\pendL	135, 136
COMMAND\pendR	136
COMMAND\pends	16
COMMAND\prev@nopbR	109
COMMAND\prev@pbR	109
COMMAND\prevpgstyle	22
COMMAND\print@Xnotes	65
COMMAND\print@Xnotes@forpages	65, 137
COMMAND\print@columnseparator	83, 136
COMMAND\print@eledsectionL	54
COMMAND\print@line	54
COMMAND\print@lineL	54
COMMAND\print@notesX@forpages	137
COMMAND\printlines	65
COMMAND\printlinesR	65, 133
COMMAND\pstart	3, 7, 10, 15–18, 28, 41, 44, 47, 48, 51, 52, 77, 81, 87, 134, 135, 138, 139
COMMAND\pstartL	52, 135
COMMAND\pstartR	52, 134, 135
COMMAND\pstartinfootnote	137
COMMAND\raw@text	77
COMMAND\read@linelist	33, 117, 138
COMMAND\ref@reg	37
COMMAND\ref@regR	37, 138
COMMAND\relax	115
COMMAND\reledmac	138
COMMAND\renewcommand	114
COMMAND\resumenumbering	27, 135
COMMAND\resumenumberingR	136
COMMAND\rightlinenumR	31, 133
COMMAND\section	133

COMMAND\section@num	24
COMMAND\selectlanguage	16, 75, 76
COMMAND\set@line	41, 138
COMMAND\set@line@action	35, 138
COMMAND\setRlineflag	16, 114
COMMAND\setgoalfraction	12, 114
COMMAND\sethangingsymbol	17
COMMAND\setline	35, 41, 138
COMMAND\setlinenum	35, 41, 138
COMMAND\setnoteposition...	84
COMMAND\setparledgroupnotespacing	114, 139
COMMAND\setposition...	84
COMMAND\setprintlines	133
COMMAND\setstanzaindents	9, 17, 114
COMMAND\setwidth...	84
COMMAND\sidenotemargin	18, 136
COMMAND\sidenotemargin*	18, 136
COMMAND\skipnumbering	15, 137
COMMAND\sloppy	8
COMMAND\stanza	7, 9, 15, 17, 46, 72, 134
COMMAND\stanzanumtrue	18
COMMAND\startlock	41, 138
COMMAND\startsub	40, 138
COMMAND\sub@action	35, 139
COMMAND\sub@off	70
COMMAND\sub@on	70
COMMAND\subline@numR	32
COMMAND\sublinenumberstyle	15, 115
COMMAND\sublinenumberstyle*	115
COMMAND\sublinenumberstyleR	15, 115
COMMAND\sublinenumincrement	15, 135, 139
COMMAND\sublinenumrepR	31, 133
COMMAND\sza@0@	17
COMMAND\textheight	13
COMMAND\textwidth	45
COMMAND\thefootnoteX	135
COMMAND\theledlanguageL	75, 76
COMMAND\theledlanguageR	75, 76
COMMAND\thepage	19, 105
COMMAND\thepstartL	15, 134
COMMAND\thepstartR	15, 134
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	42
COMMAND\xspace	20
COMMAND\xxxfootstart	84
COMMAND\xxxmatter	105
ENVIRONMENTLeftside	46
ENVIRONMENTRightside	46
ENVIRONMENTastanza	17, 72, 73, 115, 139
ENVIRONMENTledgroup	6
ENVIRONMENTpages	45
ENVIRONMENTpairs	45
PACKAGEEDMAC	115
PACKAGEEDSTANZA	115
PACKAGEEledmac	41, 65, 137
PACKAGEEledpar	137
PACKAGETABMAC	115
PACKAGEbabel	16, 17, 75, 76
PACKAGEedmac	115
PACKAGEeledmac	4, 77, 113–115, 135, 136, 138
PACKAGEeledpar	5, 13, 30, 114, 115, 135–137
PACKAGEetoolbox	83, 108
PACKAGEledmac	5
PACKAGEledpar	1, 5
PACKAGEMemoir	115
PACKAGEMusixtex	135
PACKAGEpolyglossia	16, 75, 76
PACKAGEreledmac	1, 3, 5–7, 9, 12, 13, 15–20, 22, 25, 26, 29, 30, 32, 33, 35–44, 54, 61, 70, 72, 89, 94, 114, 115, 138, 139
PACKAGEreledpar	1, 3, 5–7, 9–12, 17–22, 28, 32, 38, 39, 41, 43, 44, 70, 114, 115, 138
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
\affixpstart@numL	1
\affixpstart@numR	1
\affixside@noteR	1
\aftercolumnseparator	1, 9
\araw@textfalse	1
\araw@texttrue	1
astanza (environment)	17
\AtBeginPairs	1, 8
\AtEveryPstartCall	1

\autopar 16

B

\bbl@set@language 1
\beforecolumnseparator 1, 9
\beginnumbering 14
\beginnumberingR 1

C

\c@firstlinenumR 1
\c@firstsublinenumR 1
\c@linenumincrementR 1
\c@sublinenumincrementR 1
\ch@ck@l@ckR 1
\ch@cksub@l@ckR 1
\chapter 1
\chapterinpages 1
\check@goal 1
\check@pstarts 1
\checkpageL 1
\checkpageR 1
\checkpb@columns 1
\checkpbL 1
\checkpbR 1
\checkraw@text 1
\checkverseL 1
\checkverseR 1
\clearl@dleftpage 1
\clearl@drightpage 1
\cleartoevenpage 1
\cleartol@devenpage 1
\columnrulewidth 1, 8
\Columns 1, 8
\columns@position 1
\Columns@print@after@pend 1
\Columns@print@before@pstart 1
\columnseparator 1, 8
\columnsposition 1, 9
\correct@footinsX@box 1
\correct@Xfootins@box 1
\countLline 1
\countRline 1
\critext 1

D

\do@actions@fixedcodeR 1
\do@actions@nextR 1
\do@actionsR 1
\do@ballastR 1
\do@insidelineLhook 1

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

E

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

F

<code>\f@x@l@cksR</code>	1
<code>\finish@Pages@notes</code>	1
<code>\first@linenum@out@Rfalse</code>	1
<code>\first@linenum@out@Rtrue</code>	1
<code>\firstlinenum</code>	1, 15
<code>\firstlinenum*</code>	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

\footnoteXnomk 13

G

\get@nextboxL 1
 \get@nextboxR 1
 \getline@numR 1
 \getlinesfrompagelistL 1
 \getlinesfrompagelistR 1
 \getlinesfromparlistL 1
 \getlinesfromparlistR 1
 \goalfraction 1

H

\hidenumbering 15

I

\if@getnextbox 1
 \if@pstarts 1
 \ifaraw@text 1
 \iffirst@linenum@out@R 1
 \ifinstanzaL 1
 \ifinstanzaR 1
 \ifl@dpagfull 1
 \ifl@dpaging 1
 \ifl@dpairing 1
 \ifl@dsamepage 1
 \ifl@dusedbabel 1
 \ifledRcol 1
 \ifnomaxlines 1
 \ifnosyncpstarts 1
 \ifPages@mainmatter 1
 \ifprevpgnotnumbered 1
 \ifprint@last@after@pendL 1
 \ifprint@last@after@pendR 1
 \ifpst@rtedL 1
 \ifpst@rtedR 1
 \ifpstartnumR 1
 \ifsameparallelpagenunder 1
 \ifshiftedpstarts 1
 \ifwidthliketwocolumns 1
 \ifwrittenlinesL 1
 \init@series@par 1
 \initnumbering@sectcountR 1
 \insert@countR 1
 \insert@noterule@ledgroup 1
 \inserthangingsymbolL 1
 \inserthangingsymbolR 1
 \insertlines@listR 1
 \inserts@listR 1

L

<code>\led@set</code>	1
<code>\ledbfnote</code>	1
<code>\ledc@maxchunks</code>	1
<code>\ledcalc@maxoftwo</code>	1
<code>\ledcalc@minoftwo</code>	1
<code>\ledcalcnum</code>	1
<code>\ledchecklang</code>	1
<code>\ledleftbox</code>	1
<code>\ledlinenumR</code>	1
<code>\ledmake@labelsR</code>	1
<code>\ledminpagelines</code>	1
<code>\lednumpstartsL</code>	1
<code>\lednumpstartsR</code>	1
<code>\ledpagefullfalse</code>	1
<code>\ledpagefulltrue</code>	1
<code>\ledrightbox</code>	1
<code>\ledsamepagefalse</code>	1
<code>\ledsamepagetrue</code>	1
<code>\ledsetupmaxlinecounts</code>	1
<code>\ledsetuprawboxes</code>	1
<code>\ledskipversenumberR</code>	1
<code>\ledusedbabelfalse</code>	1
<code>\ledusedbabeltrue</code>	1
<code>\leduselanguage</code>	1
<code>\ledzeromaxlinecounts</code>	1
<code>\l@psL</code>	1
<code>\l@psR</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@memppnum</code>	1
<code>\led@error@fail@patch@outputpage</code>	1
<code>\led@error@fail@patch@pagenumbering</code>	1
<code>\led@error@fail@patch@thepage</code>	1
<code>\led@nopbnumR</code>	1
<code>\led@nopbR</code>	1
<code>\led@pbnumR</code>	1
<code>\led@pbR</code>	1
<code>\led@warn@ChangeSyncOption</code>	1
<code>\lednopbnum</code>	1
<code>\lednopbnumR</code>	1

<code>\ledpbnumR</code>	1
<code>\ledpbR</code>	1
<code>\ledstrutL</code>	1
<code>\ledstrutR</code>	1
<code>\ledthegoal</code>	1
<code>\leftlinenumR</code>	1
<code>\leftpstartnumL</code>	1
<code>\leftpstartnumR</code>	1
<code>Leftside (environment)</code>	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>\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>\setnote@position@liketwocolumns@C</code>	1
<code>\setnote@position@liketwocolumns@L</code>	1
<code>\setnote@position@liketwocolumns@R</code>	1
<code>\setposition@liketwocolumns@C</code>	1
<code>\setposition@liketwocolumns@L</code>	1
<code>\setposition@liketwocolumns@R</code>	1
<code>\setRlineflag</code>	16
<code>\setwidth@liketwocolumns@C</code>	1
<code>\setwidth@liketwocolumns@L</code>	1
<code>\setwidth@liketwocolumns@R</code>	1
<code>\sidenote@marginR</code>	1
<code>\sidenotemargin*</code>	1
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 15
<code>\startlock</code>	1
<code>\startsub</code>	1
<code>\sub@action</code>	1
<code>\subline@numR</code>	1
<code>\sublinenumberstyle*</code>	1, 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	90
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	57
\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	40
\ifledRcol: Moved \ifl@dpairing toeledmac	22
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac	25
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	31
\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	90
Added \ledstrutR to \Pages	92
\printlinesR: Simplified \printlinesR by using \setprintlines	65
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	46
\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	93
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>\iflledRcol</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	102
<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<section></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.	72
<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>	50
<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@dcnote</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.	50
\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.	70
Add \sidenotemargin*	70
\theledlanguageR: Correct left/right language setting with polyglossia.	76
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	40
\lineation*: Add \lineation*	28
\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.	76
v1.9.1.	
\ifledRcol: Moved \ifl@dpadding 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.	72
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.	65
Correct \correct@footinsX@box and \correct@Xfootins@box	65
\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	57
\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.	36
\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.	41
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.	36
\@set: \@set defined only in reledmac.	35
General: \@nl is now defined only in reledmac.	33
\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.	35
\endlock: \startlock and \endlock defined only in reledmac.	41
\endsub: \startsub and \endsub defined only in reledmac.	40
\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.	45
\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.	33
\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.	76
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 \ledstrutL and \ledstrutR</code>	95
Fix bug with dotted letter	95