

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). 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 General	5
3 Parallel columns	6
3.1 Basic use	6
3.2 Setting	7
3.2.1 Column’s width	7
3.2.2 Column’s separator	7
3.2.3 Column’s positions	7
3.2.4 Mixing two columns and one column texts	8

*This file (`reledpar.dtx`) has version number v2.2.1, last revised 2015/08/13.

[†]maieul at maieul dot net

[‡]herries dot press at earthlink dot net

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

IV Line counting	24
IV.1 Setting lineation reset	24
IV.2 Setting line number margin	25
IV.3 Setting lineation start and step	26
IV.4 Setting line flag	27
IV.5 Setting line number style	27
IV.6 Print marginal line number	27
IV.7 Line-number counters and lists	28
IV.7.1 Correspond to those in <code>reledmac</code> for regular or left text	28
IV.7.2 Specific to <code>reledpar</code>	28
IV.8 Reading the line-list file	29
IV.9 Commands within the line-list file	29
IV.10 Writing to the line-list file	34
V Marking text for notes	36
V.1 Specific hooks and commands for notes	36
V.1.1 Notes to be printed on one side only	37
V.1.2 Familiar footnotes without marks	37
V.1.3 Create hooks	38
V.1.4 Init standards series (A,B,C,D,E,Z)	39
VI Pstart numbers dumping and restoration	39
VII Parallel environments	40
VIII Paragraph decomposition and reassembly	42
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	42
VIII.2 Processing one line	47
VIII.3 Line and page number computation	51
VIII.4 Line number printing	54
VIII.5 Pstart number printing in side	56
VIII.6 Add insertions to the vertical list	58
VIII.7 Penalties	58
VIII.8 Printing leftover notes	59
IX Footnotes	60
IX.1 Line number printing	60
IX.2 Footnotes output specific to <code>\Pages</code>	60
X Cross referencing	65
XI Side notes	65
XII Familiar footnotes	67
XIII Verse	67

XIV Naming macros	69
XV Fixing babel and polyglossia	70
XVI Counts and boxes for parallel texts	72
XVII Checking text to be processed	73
XVIII Parallel columns	74
XIX Parallel pages	82
XIX.1 Specific counters	82
XIX.2 Main macro	82
XIX.3 Ensure all notes be printed at the end of parallel pages	88
XIX.4 Struts	89
XIX.5 Page clearing	89
XIX.6 Lines managing	90
XIX.7 Page break managing	91
XIX.8 Getting boxes content	94
XX Page numbering	97
XXI Sections' titles' commands	98
XXII Page break/no page break, depending on the specific line	99
XXIII Parallel ledgroup	100
XXIV Compatibility with eledmac	104
XXV The End	104
Appendix A Some things to do when changing version	105
Appendix A.1 Migration to eledpar 1.4.3	105
Appendix A.2 Migration from eledpar to reledpar	105
Appendix A.2.1 Deprecated options	105
Appendix A.2.2 \renewcommandreplaced with command	105
Appendix A.2.3 Commands the names of which have changed	106
Appendix A.3 Migration to reledpar 2.2.0	106
References	106
Index	106
Change History	123

1 Introduction

1.1 Aim of this package

Some critical editions contain texts in more than one form, such as a set of verses in one language and their translations in another. In such cases there is a desire to be able to typeset the two texts, together with any critical apparatus, in parallel. The `reledpar` package is an extension to `reledmac` that enables two texts and their apparatus to be set in parallel, either in two columns or on pairs of facing pages.

The package has to try and coerce \TeX into paths it was not designed for. Use of the package, therefore, may produce some surprising results. In this case, please reports them to the author via github's issues: <https://github.com/maieul/ledmac/issues/>.

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

You do not need to read the source code for this package in order to use it but doing so may help to answer any questions you might have. The documentation's sections are numbered in roman numeral.

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

1.2 Historical overview

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

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

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

2 General

A file may mix *numbered* and *unnumbered* text. Numbered text is printed with marginal line numbers and can include footnotes and endnotes that are referenced to those line numbers: this is how you will want to print the text that you are editing. Unnumbered text is not printed with line numbers, and you can't use `reledmac`'s note commands with it: this is appropriate for introductions and other material added by the editor around the edited text.

The `reledpar` package lets you typeset two *numbered* texts in parallel¹. 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

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

they are called out — that is, footnotes belonging to the left are set at the foot of a left (even numbered) page, and those for right texts are at the bottom of the relevant right (odd numbered) page. However, in the columnar case, all footnotes are set at the bottom left of the page on which they are called out — they are not set below the relevant column. The line numbering schemes need not be the same for the two texts.

`reledmac` essentially puts each chunk of numbered text (the text within a `\pstart ...\pend`) into a box and then following the `\pend` extracts the text line by line from the box to number and print it. More precisely, the text is first put into the box as though it was being typeset as normal onto a page and any notes are stored without being typeset. Then each typeset line is extracted from the box and any notes for that line are recalled. The line, with any notes, is then output for printing, possibly with a line number attached. Effectively, all the text is typeset and then afterwards all the notes are typeset.

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

`\maxchunks`

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

```
\maxchunks{5120}
```

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

If you `\maxchunks` is too little you can get a `reledpar` error message along the lines: “Too many `\pstart` without printing. Some text will be lost.” then you will have to either increase `\maxchunks` or use the parallel printing commands (`\Columns` or `\Pages`) more frequently.

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

In general, `reledmac` is a \TeX resource hog, and `reledpar` only makes things worse in this respect.

3 Parallel columns

3.1 Basic use

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

`\Columns`

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

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

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

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

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

There is no required pagebreak before or after the columns.

3.2 Setting

3.2.1 Column's width

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

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

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

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

3.2.2 Column's separator

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

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

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

3.2.3 Column's positions

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

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

`\beforecolumnseparator`
`\aftercolumnseparator`

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

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

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

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

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

3.2.4 Mixing two columns and one column texts

`\widthliketwocolumns`

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

`\Xnoteswidthliketwocolumns`
`\notesXwidthliketwocolumns`

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

4 Facing pages

4.1 Basic usage

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

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

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

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

4.2 Setting

4.2.1 Text width

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

`\Rcolwidth`

4.2.2 Page number

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

However, you can use the package option `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 10 p. 16.

4.2.3 Page breaking

`\setgoalfraction` When doing parallel pages `reledpar` has to guess where \TeX is going to put pagebreaks and hopefully get there first in order to put the pair of texts on their proper pages. When it thinks that the fraction `\@goalfraction` of a page has been filled, it finishes that page and starts on the other side’s text. The standard value is 0.9.

If you think you can get more on a page, increase this. On the other hand, if some left text overflows onto an odd numbered page or some right text onto an even page, try reducing it. You can change it using `\setgoalfraction{<newvalue>}`.

4.2.4 Right page before \Pages

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

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

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

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

4.3 Critical and familiar footnotes

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

4.3.1 Notes height setting

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

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

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

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

4.3.2 Notes for one side only

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

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

`\footnoteXnomk` As often happens, the left side has less room for text. We may want to call familiar notes 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 ...
```

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

```

\pend
\endnumbering
\end{Rightside}

```

5 Left and right texts

5.1 Environments

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

Leftside 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.

5.2 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.

5.3 Line numbering scheme

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

5.4 Lineation system

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

Generally speaking, controls like `\firstlinenum` or `\linenummargin` apply to sequential and left texts. To effect right texts only, they have to be within a `Rightside` environment.

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

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

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

5.5 Chunks

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

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

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

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

5.6 \AtEveryPstart and \AtEveryPstartCall

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

Consequently:

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

5.7 Language setting

If you are using the `babel` package or the `polyglossia` package, with different languages (via, say, `\selectlanguage`) for the left and right texts it is particularly important to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default. Also, it is the *last* language setting in a side that controls the language used in any notes for that side when they get printed. If you are using multilingual notes then it is probably safest to explicitly specify the language(s) for each note rather than relying on the

language selection for the side. The right side language is also applied to the right side line numbers.

5.8 Shifting

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

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

6 Verse

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

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

`\thestanzaR`

7 Side notes

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

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

8 Parallel ledgroups

8.1 General

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

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

See the following example:

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

```

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

```

8.2 Parallel ledgroups and setspace package

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

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

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

9 Sectioning commands

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

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

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

10 Notes about page number

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

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

1. If you modify `\thepage` command, use the value of `par@page` counter inside and not the value of `page` counter.

2. If you want to modify a page number, modify the value of page counter AND the value `par@page` counter.

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

I Implementation overview

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

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

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

II Preliminaries

II.1 Package’s meta-data

Announce the name and version of the package, which is targeted for $\text{\LaTeX}2\text{e}$. 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}[2015/08/13 v2.2.1 reledmac extension for
  parallel texts]%
4
5 %
```

II.2 Package’s requirement

Few commands use `\xspace` command.

```

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

II.3 Package’s options

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

```

8 \RequirePackage{xkeyval}
9 %
```

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

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

The parledgroup can be called either on reledmac or reledpar.

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

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

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

`\ifsameparallelpagenumber` Options related to page numbering

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

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

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

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

II.4 Determining side and category of parallel processing

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

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

II.5 Text's width

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

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

II.6 Messages

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

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

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

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

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

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

```

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

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

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

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

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

```

III Sectioning commands

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

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

```

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

```

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

```

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

```

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

```

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

```

\endnumbering This is the left text version of the regular `\endnumbering` and must follow the last text for a left text numbered section. It sets `\ifpst@rtedL` to FALSE. It is fully defined in `reledmac`.

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

```

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

```

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

```

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

```

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

`\resumenumberingR`

```

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

```

```

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

```

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

```

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

```

IV Line counting

IV.1 Setting lineation reset

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

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

```

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

```

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

```

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

```

IV.2 Setting line number margin

`\linenummargin` You call `\linenummargin{⟨word⟩}` to specify which margin you want your right text's line numbers in; it takes one argument, a string. You can put the line numbers in the

`\line@marginR`

same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. You can change this within a numbered section, but the change may not take effect just when you would like; if it is done between paragraphs nothing surprising should happen.

For right texts the selection is recorded in the count `\line@marginR`, otherwise in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.

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

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

By default put right text numbers at the right.

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

IV.3 Setting lineation start and step

`\c@firstlinenumR`
`\c@linenumincrementR`

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

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

`\c@firstsublinenumR`
`\c@sublinenumincrementR`

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

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

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

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

```
244 \WithSuffix\newcommand\firstlinenum*[1]{%
245 \setcounter{firstlinenumR}{#1}%
246 \setcounter{firstlinenum}{#1}%
247 }
248 \WithSuffix\newcommand\linenumincrement*[1]{%
249 \setcounter{linenumincrementR}{#1}%
```

```

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

```

IV.4 Setting line flag

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

```

261 \newcommand{\setRlineflag}[1]{%
262 \gdef\Rlineflag{#1}%
263 }
264 \setRlineflag{R}
265 %

```

IV.5 Setting line number style

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

```

266 \newcommand*\linenumrepR[1]{\@arabic{#1}}
267 \newcommand*\sublinenumrepR[1]{\@arabic{#1}}
268
269 %

```

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`.

```

270 \newcommand*\leftlinenumR{%
271 \l@dlinenumR
272 \kern\linenumsep}
273 \newcommand*\rightlinenumR{%
274 \kern\linenumsep
275 \l@dlinenumR}
276 \newcommand*\l@dlinenumR{%
277 \numlabfont\linenumrepR{line@numR}\Rlineflag%
278 \ifsublines@
279 \ifnum\subline@num>\z@
280 \unskip\fullstop\sublinenumrepR{\subline@numR}%

```

```

281 \fi
282 \fi}
283
284 %

```

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.

```

285 \newcount\line@numR
286 \newcount\subline@numR
287 \newcount\absline@numR
288
289 %

```

`\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`

```

290 \list@create{\line@listR}
291 \list@create{\insertlines@listR}
292 \list@create{\actionlines@listR}
293 \list@create{\actions@listR}
294
295 %

```

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

```

296 \newcount\page@numR
297
298 %

```

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`

```

299 \list@create{\linesinpar@listL}
300 \list@create{\linesinpar@listR}
301 \list@create{\maxlinesinpar@list}
302
303 %

```

IV.8 Reading the line-list file

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

```

304 \newcommand{\list@clearing@regR}{%
305   \list@clear{\line@listR}%
306   \list@clear{\insertlines@listR}%
307   \list@clear{\actionlines@listR}%
308   \list@clear{\actions@listR}%
309   \list@clear{\linesinpar@listR}%
310   \list@clear{\linesonpage@listR}
311 }
312 %

```

\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.

\@nl

```

313 \newcommand{\@nl@regR}{%
314   \ifx\l@dchset@num\relax \else
315     \advance\absline@numR \@ne
316     \set@line@action
317     \let\l@dchset@num\relax
318     \advance\absline@numR \m@ne
319     \advance\line@numR \m@ne%    % do we need this?
320   \fi
321   \advance\absline@numR \@ne
322   \ifx\next@page@numR\relax \else
323     \page@action
324     \let\next@page@numR\relax
325   \fi
326   \ifx\sub@change\relax \else

```

```

327 \ifnum\sub@change>\z@
328   \sublines@true
329 \else
330   \sublines@false
331 \fi
332 \sub@action
333 \let\sub@change\relax
334 \fi
335 \ifcase\@lockR
336 \or
337   \@lockR \tw@
338 \or\or
339   \@lockR \z@
340 \fi
341 \ifcase\sub@lockR
342 \or
343   \sub@lockR \tw@
344 \or\or
345   \sub@lockR \z@
346 \fi
347 \ifsublines@
348   \ifnum\sub@lockR<\tw@
349     \advance\subline@numR \@ne
350   \fi
351 \else
352   \ifnum\@lockR<\tw@
353     \advance\line@numR \@ne \subline@numR \z@
354   \fi
355 \fi}
356
357
358 %

```

`\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`.

```

359 \newcount\last@page@numR
360 \last@page@numR=-10000
361
362 %

```

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

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

`\ld@set` The `\ld@set{<num>}` macro sets the line number for the next `\pstart...` to the

value specified as its argument. This is used to implement `\setlinenum`. It is defined in `reledmac`.

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

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

`\sub@action` `\sub@action` adds an entry to the action-code list to turn sub-lineation on or off, according to the current value of the `\ifsublines@` flag. It is defined in `reledmac`.

`\do@lockon` `\lock@on` adds an entry to the action-code list to turn line number locking on. The 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`.

```

363 \newcount\@lockR
364 \newcount\sub@lockR
365
366 \newcommand*{\do@lockonR}{%
367   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
368   \ifsublines@
369     \xright@appenditem{-1005}\to\actions@listR
370     \ifnum\sub@lockR=\z@
371       \sub@lockR \@ne
372     \else
373       \ifnum\sub@lockR=\thr@@
374         \sub@lockR \@ne
375       \fi
376     \fi
377   \else
378     \xright@appenditem{-1003}\to\actions@listR
379     \ifnum\@lockR=\z@
380       \@lockR \@ne
381     \else
382       \ifnum\@lockR=\thr@@
383         \@lockR \@ne
384       \fi
385     \fi
386   \fi}
387
388 %

```

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

`\do@lockoffR` `\do@lockoffR`.

`\skip@lockoff`

```

389
390

```

```

391 \newcommand{\do@lockoffR}{%
392 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
393 \ifsublines@
394 \xright@appenditem{-1006}\to\actions@listR
395 \ifnum\sub@lockR=\tw@
396 \sub@lockR \thr@@
397 \else
398 \sub@lockR \z@
399 \fi
400 \else
401 \xright@appenditem{-1004}\to\actions@listR
402 \ifnum\@lockR=\tw@
403 \@lockR \thr@@
404 \else
405 \@lockR \z@
406 \fi
407 \fi}
408
409
410 %

```

`\n@num`

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

`\@ref@regR`
`\insert@countR`

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

```

411 \newcount\insert@countR
412 %

```

- #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.

```

413 \newcommand*{\@ref@regR}[2]{%
414 \global\advance\@edtext@level by 1%
415 \global\insert@countR=#1\relax
416 \loop\ifnum\insert@countR>\z@
417 \xright@appenditem{\the\absline@numR}\to\insertlines@listR
418 \global\advance\insert@countR \m@ne
419 \repeat
420 %

```


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.

```

421 \begin{group
422   \let\@ref=\dummy@ref
423   \let\@lopR\@gobble
424   \let\page@action=\relax
425   \let\sub@action=\relax
426   \let\set@line@action=\relax
427   \let\@lab=\relax
428   \let\@lemma=\relax
429   \let\@sw\@gobblethree%
430   #2
431   \global\endpage@num=\page@numR
432   \global\endline@num=\line@numR
433   \global\endsubline@num=\subline@numR
434 \end{group}
435 %

```

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

```

436 \xright@appenditem%
437   {\the\page@numR|\the\line@numR|}%
438   \ifsublines@ \the\subline@numR \else 0\fi}%
439   \the\endpage@num|\the\endline@num|}%
440   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
441 %

```

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

```

442 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
443   @edtext@level\endcsname}%
444 %

```

Declare and init boolean for lemma in this level.

```

444 \providebool{lemmacommand@\the\@edtext@level}%
445 \boolfalse{lemmacommand@\the\@edtext@level}%
446 %

```

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

```

447 #2
448 % Now, we store the list of \protect\cs{@sw} of this current \protect\cs{
449   edtext} as an element of
450 % the global list of list of \protect\cs{@sw} for a \protect\cs{edtext}
451 % depth.
452 % \begin{macrocode}

```

```

451 \ifnum\@edtext@level>0%
452 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtextR@the\@edtext@level\endcsname}}%
453 \ifcsundef{sw@list@edtextR@the\@edtext@level}{\
create@this@edtext@level}{}%
454 \letcs{\@tmp}{sw@list@edtextR@the\@edtext@level}%
455 \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}%
456 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
457 \global\cslet{sw@list@edtextR@the\@edtext@level}{\@tmp}%
458 \fi%
459 %

```

Decrease edtext level counter.

```

460 \global\advance\@edtext@level by -1%
461 }
462 %

```

\@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`.

```

463 \renewcommand*{\@pend}[1]{%
464 \ifbypstart@global\line@num=0\fi%
465 \xright@appenditem{#1}\to\linesinpar@listL}
466 \renewcommand*{\@pendR}[1]{%
467 \ifbypstart@R\global\line@numR=0\fi
468 \xright@appenditem{#1}\to\linesinpar@listR}
469 %
470 %

```

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

```

471 \renewcommand*{\@lopL}[1]{%
472 \xright@appenditem{#1}\to\linesonpage@listL}
473 \renewcommand*{\@lopR}[1]{%
474 \xright@appenditem{#1}\to\linesonpage@listR}
475 %
476 %

```

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`.

```

477 \newwrite\linenum@outR
478 %

```

`\iffirst@linenum@out@R` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open.

```

\first@linenum@out@Rtrue
\first@linenum@out@Rfalse
479 \newif\iffirst@linenum@out@R
480 \first@linenum@out@Rtrue
481 %

```

`\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.

```

482 \newcommand*{\line@list@stuffR}[1]{%
483   \read@linelist{#1}%
484   \iffirst@linenum@out@R
485     \immediate\closeout\linenum@outR
486     \global\first@linenum@out@Rfalse
487     \immediate\openout\linenum@outR=#1
488     \immediate\write\linenum@outR{\string\line@list@version{\
this@line@list@version}}}%
489   \else
490     \if@minipage%
491       \leavevmode%
492       \fi%
493     \closeout\linenum@outR%
494     \openout\linenum@outR=#1%
495   \fi}
496
497 %

```

`\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.

```

498 \newcommand*{\new@lineL}{%
499   \write\linenum@out{\string\@nl[\the\c@page][\thepage]}}
500 %

```

`\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.

```

501 \newcommand*{\new@lineR}{%
502   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}
503 %

```

`\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`.

<code>\startsub</code> <code>\endsub</code>	<code>\startsub</code> and <code>\endsub</code> turn sub-lineation on and off, by writing appropriate instructions to the line-list file. There are both defined in <code>reledmac</code> .
<code>\advanceline</code>	You can use <code>\advanceline{⟨num⟩}</code> in running text to advance the current visible line-number by a specified value, positive or negative. It is defined in <code>reledmac</code> .
<code>\setline</code>	You can use <code>\setline{⟨num⟩}</code> in running text (i.e., within <code>\pstart... \pend</code>) to set the current visible line-number to a specified positive value. It is defined in <code>reledmac</code> .
<code>\setlinenum</code>	You can use <code>\setlinenum{⟨num⟩}</code> before a <code>\pstart</code> to set the visible line-number to a specified positive value. It writes a <code>\l@d@set</code> command to the line-list file. It is defined in <code>reledmac</code> .
<code>\startlock</code> <code>\endlock</code>	You can use <code>\startlock</code> or <code>\endlock</code> 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 <code>reledmac</code> .
<code>\skipnumbering</code>	

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.

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

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.

<code>\newseries@par</code>	<code>\newcommand{\newseries@par}[1]{%</code>
505	<code>%</code>

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.

```

506 \unless\ifnofamiliar%
507   \csgdef{onlysideX@#1}{}%
508   \global\newbool{keepforsideX@#1}%
509 \fi%
510 \unless\ifnocritical%
511   \global\newbool{keepforXside@#1}%
512   \csgdef{Xonlyside@#1}{}%
513 \fi%
514 %

```

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`.

```

515 \unless\ifnofamiliar%
516 % So declare the list.
517 %   \begin{macrocode}
518   \expandafter\list@create\csname footnote#1@mk\endcsname%
519 %

```

Then, declare the `\footnoteXnomk` command.

```

520 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
521 %

```

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

```

522 \toggletrue{nomk@}%
523 \csuse{footnote#1}{##1}%
524 \togglefalse{nomk@}%
525 %

```

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`.

```

526 \letcs{\@tmp}{footnote#1@mk}%
527 \numdef\@tmpa{\csuse{c{footnote#1}}}%
528 \global\xright@appenditem{\@tmpa}\to\@tmp%
529 \global\cslet{footnote#1@mk}{\@tmp}%
530 }%
531 %

```

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

```

532 \expandafter\newcommand\csname footnote#1mk\endcsname{%
533 %

```

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

```

534 \letcs{\@tmp}{footnote#1@mk}%
535 \gl@p\@tmp\to\@tmpa%
536 \global\cslet{footnote#1@mk}{\@tmp}%
537 %

```

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

```

538 \letcs{\old@footnote}{c@footnote#1}%
539 \setcounter{footnote#1}{\@tmpa}%
540 %

```

Define the footnote mark and print it

```

541 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
542 \csuse{@footnotemark#1}%
543 %

```

Restore previous footnote counter and finally add space.

```

544 \setcounter{footnote#1}{\old@footnote}%
545 \xspace%
546 }%
547 %

```

End of tools for familiar notes without marks

```

548 \fi
549 %

```

End of `\newseries@par`.

```

550 }%
551 %

```

V.1.3 Create hooks

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

```

552 \unless\ifnocritical@%
553 \newhookcommand@series{Xonlyside}%
554 \fi%
555 \unless\ifnofamiliar@%
556 \newhookcommand@series{onlysideX}%
557 \fi
558
559
560 %

```

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`.

```

561 \newcommand{\init@series@par}{%
562   \def\do##1{\newseries@par{##1}}%
563   \dolistloop{\@series}%
564 }%
565 \init@series@par%
566 %

```

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@pc  \list@create{\list@pstartL@pc}%
\list@pstartR@pc  \list@create{\list@pstartR@pc}%
569 %

```

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

```

\dump@pstartL@pc  \def\dump@pstartL@pc{%
\dump@pstartR@pc  \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
572 \global\cslet{numberpstart@L\the\l@dnumstartsL}{\ifnumberpstart}%
573 }%
574
575 \def\dump@pstartR@pc{%
576 \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
577 \global\cslet{numberpstart@R\the\l@dnumstartsR}{\ifnumberpstart}%
578 }%
579
580 %

```

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

```

\restore@pstartR@pc
581 \def\restore@pstartL@pc{%
582   \ifx\list@pstartL@pc\empty\else%

```

```

583 \gl@p\list@pstartL@pc\to\@temp%
584 \global\c@pstartL=\@temp%
585 \fi%
586 }%
587 \def\restore@pstartR@pc{%
588 \ifx\list@pstartR@pc\empty\else%
589 \gl@p\list@pstartR@pc\to\@temp%
590 \global\c@pstartR=\@temp%
591 \fi%
592 }%
593 %

```

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.

```

594 \newenvironment{pairs}{%}
595 \l@dpairingtrue
596 \l@dpagingfalse
597 \initnumbering@quote
598 \at@begin@pairs%
599 }{%
600 \l@dpairingfalse
601 }
602
603 %

```

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

```

604 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
605 \def\at@begin@pairs{%
606
607 %

```

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.

```

608 \newenvironment{pages}{%
609 \l@dpairingtrue
610 \l@dpagingtrue
611 \initnumbering@quote
612 \setlength{\Lcolwidth}{\textwidth}%
613 \setlength{\Rcolwidth}{\textwidth}%

```



```

614 }{
615   \l@dpairingfalse
616   \l@dpagingfalse
617 }
618
619 %

```

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

ifinstanzaR

```

620 \newif\ifinstanzaL
621 \newif\ifinstanzaR
622 %

```

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.

```

623 \newenvironment{Leftside}{%
624   \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
625     \led@err@Leftside@PreviousNotPrinted%
626   \fi%
627   \ledRcolfalse
628   \setcounter{pstartL}{1}
629   \let\pstart\pstartL
630   \let\thepstart\thepstartL
631   \let\pend\pendL
632   \let\memorydump\memorydumpL
633   \Leftsidehook
634   \let\old@startstanza\@startstanza
635   \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
636 }{
637   \Leftsidehookend}
638 %

```

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

\Leftsidehookend

```

639 \Rightsidehook
640 \Rightsidehookend
641 \newcommand*{\Leftsidehook}{}
642 \newcommand*{\Leftsidehookend}{}
643 \newcommand*{\Rightsidehook}{}
644 \newcommand*{\Rightsidehookend}{}
645 %

```

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.

```

645 \newenvironment{Rightside}{%
646   \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
647   \led@err@Rightside@PreviousNotPrinted%
648   \fi%
649   \ledRcoltrue
650   \let\beginnumbering\beginnumberingR
651   \let\endnumbering\endnumberingR
652   \let\pausenumbering\pausenumberingR
653   \let\resumenumbering\resumenumberingR
654   \let\memorydump\memorydumpR
655   \let\thepstart\thepstartR
656   \let\pstart\pstartR
657   \let\pend\pendR
658   \let\ledpb\ledpbR
659   \let\lednopb\lednopbR
660   \let\lineation\lineationR
661   \Rightsidehook
662   \let\old@startstanza\@startstanza
663   \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
664 }{%
665   \ledRcolfalse
666   \Rightsidehookend
667 }
668
669 %

```

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.

```

670 \newcount\num@linesR

```

```

671 \newbox\one@lineR
672 \newcount\par@lineR
673 %

```

\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.

```

674
675 \newcounter{pstartL}
676 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
677 \newcounter{pstartR}
678 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
679
680 \newcommand*{\pstartL}[1][1]{%
681   \if@nobeak%
682     \let\@oldnobeak\@nobeaktrue%
683   \else%
684     \let\@oldnobeak\@nobeakfalse%
685   \fi%
686   \@nobeaktrue%
687   \ifluatex%
688     \xdef\l@luatextextdir@L{\the\luatextextdir}%
689     \xdef\l@luatexpardir@L{\the\luatexpardir}%
690     \xdef\l@luatexbodydir@L{\the\luatexbodydir}%
691   \fi%
692   \ifnumbering \else%
693     \led@err@PstartNotNumbered%
694     \beginnumbering%
695   \fi%
696   \ifnumberedpar@%
697     \led@err@PstartInPstart%
698     \pend%
699   \fi%
700 %

```

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

```

701 \ifpst@rtedL\else%
702   \list@clear{\inserts@list}%
703   \global\let\next@insert=\empty%
704   \global\pst@rtedLtrue%
705 \fi%
706 \begingroup\normal@pars%

```

707 %

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.

```
708 \global\advance\l@dnumpstartsL \@ne%
709 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
710 \led@err@TooManyPstarts%
711 \global\l@dnumpstartsL=\l@dc@maxchunks%
712 \fi%
713 \global\setnamebox{\l@dc@l@rawbox\the\l@dnumpstartsL}=\vbox\bgroup%
714 %
```

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.

```
715 \l@dzeropenalties%
716 \ifautopar\else%
717 \ifnumberpstart%
718 \ifsidepstartnum%
719 \else%
720 \thepstartL%
721 \fi%
722 \fi%
723 \fi%
724 \hsize=\Lcolwidth%
725 \numberedpar@true%
726 \iflabelpstart\protected@edef\@currentlabel%
727 {\p@pstartL\thepstartL}\fi%
728 %
```

Dump the optional arguments

```
729 \ifstrempy{#1}%
730 {\csgdef{before@pstartL@the\l@dnumpstartsL}{\at@every@pstart}}%
731 {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent#1}}%
732 \at@every@pstart@call%
733 %
```

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

```
734 \ignorespaces%
735 %
```

```
736 }
737 %
```

The same for right side.

```
738 \newcommand*{\pstartR}[1][1]{%
739 \if@nbreak%
740 \let\@oldnbreak\@nbreaktrue%
741 \else%
```

```

742 \let\@oldnbreak\@nbreakfalse%
743 \fi%
744 \@nbreaktrue%
745 \ifluatex%
746 \xdef\l@luatextextdir@R{\the\luatextextdir}%
747 \xdef\l@luatexpardir@R{\the\luatexpardir}%
748 \xdef\l@luatexbodydir@R{\the\luatexbodydir}%
749 \fi%
750 \ifnumberingR \else%
751 \led@err@PstartNotNumbered%
752 \beginnumberingR%
753 \fi%
754 \ifnumberedpar@%
755 \led@err@PstartInPstart%
756 \pendR%
757 \fi%
758 \ifpst@rtedR\else%
759 \list@clear{\inserts@listR}%
760 \global\let\next@insertR=\empty%
761 \global\pst@rtedRtrue%
762 \fi%
763 \begingroup\normal@pars%
764 \global\advance\l@dnumpstartsR \@ne%
765 \ifnum\l@dnumpstartsR>\l@dc@maxchunks%
766 \led@err@TooManyPstarts%
767 \global\l@dnumpstartsR=\l@dc@maxchunks%
768 \fi%
769 \global\setnamebox{\l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
770 \l@dzeropenalties%
771 \ifautopar\else%
772 \ifnumberpstart%
773 \ifsidepstartnum\else%
774 \thepstartR%
775 \fi%
776 \fi%
777 \fi%
778 \hsize=\Rcolwidth%
779 \numberedpar@true%
780 \iflabelpstart\protected@edef\@currentlabel%
781 {\p@pstartR\thepstartR}\fi%
782 \ifstrempy{#1}%
783 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
784 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
785 \at@every@pstart@call%
786 \ignorespaces%
787 }
788 %

```

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

```

789 \newcommandx*{\pendL}[1][1]{%
790   \ifnumbering \else%
791     \led@err@PendNotNumbered%
792   \fi%
793   \ifnumberedpar@ \else%
794     \led@err@PendNoPstart%
795   \fi%
796 %

```

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.

```

797   \endgraf\global\num@lines=\prevgraf\egroup%
798   \global\par@line=0%
799 %

```

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

```

800   \endgroup%
801   \ignorespaces%
802   \@oldnbreak%
803   \dump@pstartL@pc%
804   \ifnumberpstart%
805     \addtocounter{pstartL}{1}%
806   \fi
807   \parledgroup@beforenotes@save{L}%
808 %

```

Dump content of the optional argument.

```

809   \ifstrempy{#1}%
810     {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
811     {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
812   }
813 %

```

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

```

814 \newcommandx*{\pendR}[1][1]{%
815   \ifnumberingR \else%
816     \led@err@PendNotNumbered%
817   \fi%
818   \ifnumberedpar@ \else%
819     \led@err@PendNoPstart%
820   \fi%
821   \endgraf\global\num@linesR=\prevgraf\egroup%
822   \global\par@lineR=0%
823   \endgroup%
824   \ignorespaces%
825   \@oldnbreak%
826   \dump@pstartR@pc%
827   \ifnumberpstart%

```

```

828 \addtocounter{pstartR}{1}%
829 \fi%
830 \parledgroup@beforenotes@save{R}%
831 \ifstrempy{#1}%
832   {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
833   {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
834 }
835
836 %

```

\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.

```

837 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call{#1}}%
838 \gdef\at@every@pstart@call{}%
839 %

```

\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

```

840 \newif\ifprint@last@after@pendL%
841 \newif\ifprint@last@after@pendR%
842 %

```

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

```

843 \newbox\l@dleftbox
844 \newbox\l@drightbox
845
846 %

```

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

\countRline

```

847 \newcount\countLline
848 \countLline \z@
849 \newcount\countRline
850 \countRline \z@
851
852 %

```

\@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

```

853 \newcount\@donereallinesL
854 \newcount\@donetotallinesL
855 \newcount\@donereallinesR
856 \newcount\@donetotallinesR
857
858 %

```

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

```

859 \newcommand*\do@lineL{%
860   \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
861   \advance\countLline \@ne%
862   \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
863     {\vbadness=10000%
864     \splittopskip=\z@%
865     \do@lineLhook%
866     \l@demptyd@ta%
867     \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}%
868       to\baselineskip}%
869   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startL
870 }{%
871   \unvbox\one@line \global\setbox\one@line=\lastbox%
872   \getline@numL%
873   \ifnum\@lock>\@ne%
874     \inserthangingsymboltrue%
875   \else%
876     \inserthangingsymbolfalse%
877   \fi%
878   \setbox\l@dleftbox%
879   \hb@xt@ \Lcolwidth{%
880     \ifl@dhidenumber%
881       \global\l@dhidenumberfalse%
882       \f@x@l@cks%
883     \else%
884       \affixline@num%
885     \fi%
886     \xifinlist{\the\l@dpscl}{\eled@sections@}%
887     {\add@inserts\affixside@note}%
888     {\print@lineL}}%
889   \add@penaltiesL%
890   \global\advance\@donereallinesL\@ne%
891   \global\advance\@donetotallinesL\@ne%
892 \else%
893   \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
894   \global\advance\@donetotallinesL\@ne%
895 \fi}
896
897 %

```


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

```

898 \def\print@lineL{%
899   \affixpstart@numL%
900   \l@dld@ta %space kept for backward compatibility
901   \add@inserts\affixside@note%
902   \l@dlsn@te %space kept for backward compatibility
903   {\ledllfill\hb@xt@ \Lcolwidth{%
904     \do@insidelineLhook%
905     \ifluatex%
906       \luatextextdir\l@luatextextdir@L%
907     \fi%
908     \new@lineL%
909     \inserthangingsymbolL%
910     \l@dunhbox@line{\one@line}}\ledrlfill\l@drd@ta%
911   \l@drsn@te}}
912
913 %

```

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

```

914 \def\print@eledsectionL{%%
915   \addtocounter{pstartL}{-1}%
916   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
917   \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}%
918   \numdef{\temp@}{\l@dpscL-1}%
919   \xifinlist{\temp@}{\eled@sections@}{\@nbreaktrue}{\@nbreakfalse}%
920   \@eled@sectioningtrue%
921   \bgroup%
922     \ifluatex%
923       \luatextextdir\l@luatextextdir@L%
924       \luatexpardir\l@luatexpardir@L%
925       \luatexbodydir\l@luatexbodydir@L%
926       \ifdefstring{\l@luatextextdir@L}{TRT}{\@RTLtrue}{}%
927     \fi%
928     \csuse{eled@sectioning@the\l@dpscL}%
929   \egroup%
930   \@eled@sectioningfalse%
931   \global\csundef{eled@sectioning@the\l@dpscL}%
932   \if@RTL%
933     \hspace{-3\paperwidth}%
934     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
935   \else%
936     \hspace{3\paperwidth}%
937     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
938   \fi%
939   \vskip\eledsection@correcting@skip%
940 }
941
942 %

```

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

```

\doinssideLhook 943 \newcommand*{\dolineLhook}[1]{\gdef\do@lineLhook{#1}}%
\doinssideRhook 944 \newcommand*{\dolineRhook}[1]{\gdef\do@lineRhook{#1}}%
945 \newcommand*{\doinssideLhook}[1]{\gdef\do@insidelineLhook{#1}}%
946 \newcommand*{\doinssideRhook}[1]{\gdef\do@insidelineRhook{#1}}%
947
948 %

```

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

```

\dolineRhook 949 \newcommand*{\do@lineLhook}{}
\doinssideLhook 950 \newcommand*{\do@lineRhook}{}
\doinssideRhook 951 \newcommand*{\do@insidelineLhook}{}
952 \newcommand*{\do@insidelineRhook}{}
953
954 %

```

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

```

955 \newcommand*{\do@lineR}{%
956   \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscr}%
957   \ledRcol@true%
958   \advance\countRline \@ne%
959   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscr}%
960   {\vbadness=10000%
961    \splittopskip=\z@%
962    \do@lineRhook%
963    \l@demtyd@ta%
964    \global\setbox\one@lineR=\vsplit\namebox{1@dRcolrawbox\the\l@dpscr}%
965    to\baselineskip}%
966   \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\parledgroup@notes@startR
967   }{}%
967   \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
968   \getline@numR%
969   \ifnum\@lockR>\@ne%
970     \inserthangingsymbolRtrue%
971   \else%
972     \inserthangingsymbolRfalse%
973   \fi%
974   \setbox\l@drightbox%
975   \hb@xt@ \Rcolwidth{%
976     \ifl@dhidenumber%
977       \global\l@dhidenumberfalse%
978       \f@x@l@cksR%
979     \else%
980       \affixline@numR%
981     \fi%

```

```

982 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
983 {\add@insertsR\affixside@noteR}%
984 {\print@lineR}%
985 }%
986 \add@penaltiesR%
987 \global\advance\@donereallinesR\@ne%
988 \global\advance\@donetotallinesR\@ne%
989 \else%
990 \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}%
991 \global\advance\@donetotallinesR\@ne%
992 \fi%
993 \ledRcol@false%
994 }
995
996
997 %

```

`\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.

```

998 \newcommand*{\getline@numR}{%
999 \global\advance\absline@numR \@ne
1000 \do@actionsR
1001 \do@ballastR
1002 \ifledgroupnotesR\else
1003 \ifnumberline
1004 \ifsublines@
1005 \ifnum\sub@lockR<\tw@
1006 \global\advance\subline@numR \@ne
1007 \fi
1008 \else
1009 \ifnum\@lockR<\tw@
1010 \global\advance\line@numR \@ne
1011 \global\subline@numR \z@
1012 \fi
1013 \fi
1014 \fi
1015 \fi
1016 }
1017 \newcommand*{\getline@numL}{%
1018 \global\advance\absline@num \@ne
1019 \do@actions
1020 \do@ballast
1021 \ifledgroupnotesL\else
1022 \ifnumberline
1023 \ifsublines@

```

```

1024         \ifnum\sub@lock<\tw@
1025             \global\advance\subline@num \@ne
1026         \fi
1027     \else
1028         \ifnum\@lock<\tw@
1029             \global\advance\line@num \@ne
1030             \global\subline@num \z@
1031         \fi
1032     \fi
1033 \fi
1034 \fi
1035 }
1036
1037
1038 %

```

`\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.

```

1039 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1040 \begingroup
1041     \advance\absline@numR \@ne
1042     \ifnum\next@actionlineR=\absline@numR
1043         \ifnum\next@actionR>-1001
1044             \global\advance\ballast@count by -\c@ballast
1045         \fi
1046     \fi
1047 \endgroup}
1048 %

```

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

```

1049
1050 \newif\ifl@dskipversenumberR
1051 \newcommand*{\do@actions@fixedcodeR}{%
1052     \ifcase\@l@dttempcnta%
1053     \or% % 1001
1054         \global\sublines@true
1055     \or% % 1002
1056         \global\sublines@false
1057     \or% % 1003
1058         \global\@lockR=\@ne
1059     \or% % 1004%
1060         \ifnum\@lockR=\tw@
1061             \global\@lockR=\thr@@
1062         \else
1063             \global\@lockR=\z@

```

```

1064 \fi
1065 \or% % 1005
1066 \global\sub@lockR=\@ne
1067 \or% % 1006
1068 \ifnum\sub@lockR=\tw@
1069 \global\sub@lockR=\thr@@
1070 \else
1071 \global\sub@lockR=\z@
1072 \fi
1073 \or% % 1007
1074 \l@dskipnumbertrue
1075 \or% % 1008
1076 \l@dskipversenumberRtrue%
1077 \or% % 1009
1078 \l@dhiddenumbertrue%
1079 \else%
1080 \led@warn@BadAction
1081 \fi%
1082 }
1083
1084
1085 \newcommand*{\do@actionsR}{%
1086 \global\let\do@actions@nextR=\relax
1087 \@l@dttempcntb=\absline@numR
1088 \ifnum\@l@dttempcntb<\next@actionlineR\else
1089 \ifnum\next@actionR>-1001\relax
1090 \global\page@numR=\next@actionR
1091 \ifbypage@R
1092 \global\line@numR \z@ \global\subline@numR \z@
1093 \fi
1094 \else
1095 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1096 \@l@dttempcnta=-\next@actionR
1097 \advance\@l@dttempcnta by -5001\relax
1098 \ifsublines@
1099 \global\subline@numR=\@l@dttempcnta
1100 \else
1101 \global\line@numR=\@l@dttempcnta
1102 \fi
1103 \else
1104 \@l@dttempcnta=-\next@actionR
1105 \advance\@l@dttempcnta by -1000\relax
1106 \do@actions@fixedcodeR
1107 \fi
1108 \fi
1109 \ifx\actionlines@listR\empty
1110 \gdef\next@actionlineR{1000000}%
1111 \else
1112 \gl@p\actionlines@listR\to\next@actionlineR
1113 \gl@p\actions@listR\to\next@actionR

```

```

1114 \global\let\do@actions@nextR=\do@actionsR
1115 \fi
1116 \fi
1117 \do@actions@nextR}
1118
1119 %

```

VIII.4 Line number printing

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

```

1120 \ch@cksub@l@ckR
1121 \ch@ck@l@ckR
1122 \f@x@l@cksR
1123 \affixline@numR
1124 \newcommand*{\l@dcalcnum}[3]{%
1125 \ifnum #1 > #2\relax
1126 \l@dttempcnta = #1\relax
1127 \advance\l@dttempcnta by -#2\relax
1128 \divide\l@dttempcnta by #3\relax
1129 \multiply\l@dttempcnta by #3\relax
1130 \advance\l@dttempcnta by #2\relax
1131 \else
1132 \l@dttempcnta=#2\relax
1133 \fi}
1134
1135 \newcommand*{\ch@cksub@l@ckR}{%
1136 \ifcase\sub@lockR
1137 \or
1138 \ifnum\sublock@disp=\@ne
1139 \l@dttempcntb \z@ \l@dttempcnta \@ne
1140 \fi
1141 \or
1142 \ifnum\sublock@disp=\tw@
1143 \else
1144 \l@dttempcntb \z@ \l@dttempcnta \@ne
1145 \fi
1146 \or
1147 \ifnum\sublock@disp=\z@
1148 \l@dttempcntb \z@ \l@dttempcnta \@ne
1149 \fi
1150 \fi}
1151
1152 \newcommand*{\ch@ck@l@ckR}{%
1153 \ifcase\@lockR
1154 \or
1155 \ifnum\lock@disp=\@ne
1156 \l@dttempcntb \z@ \l@dttempcnta \@ne
1157 \fi
1158 \or
1159 \ifnum\lock@disp=\tw@
1160 \else
1161 \l@dttempcntb \z@ \l@dttempcnta \@ne
1162 \fi
1163 \or
1164 \ifnum\lock@disp=\z@
1165 \l@dttempcntb \z@ \l@dttempcnta \@ne
1166 \fi
1167 \fi}

```

```

1158 \l@tempcntb \z@ \l@tempcnta \@ne
1159 \fi
1160 \or
1161 \ifnum\lock@disp=\z@
1162 \l@tempcntb \z@ \l@tempcnta \@ne
1163 \fi
1164 \fi}
1165
1166 \newcommand*{\f@x@l@cksR}{%
1167 \ifcase\@lockR
1168 \or
1169 \global\@lockR \tw@
1170 \or \or
1171 \global\@lockR \z@
1172 \fi
1173 \ifcase\sub@lockR
1174 \or
1175 \global\sub@lockR \tw@
1176 \or \or
1177 \global\sub@lockR \z@
1178 \fi}
1179
1180
1181 \newcommand*{\affixline@numR}{%
1182 \ifledgroupnotesR\else\ifnumberline
1183 \ifl@skipnumber
1184 \global\l@skipnumberfalse
1185 \else
1186 \ifsublines@
1187 \l@tempcntb=\subline@numR
1188 \l@dcalcnum{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR
1189 }%
1190 \ch@cksub@lockR
1191 \else
1192 \l@tempcntb=\line@numR
1193 \ifx\linenumberlist\empty
1194 \l@dcalcnum{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1195 \else
1196 \l@tempcnta=\line@numR
1197 \edef\rem@inder{\linenumberlist,\number\line@numR,}%
1198 \edef\sc@n@list{\def\noexpand\sc@n@list
1199 ###1,\number\l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1200 \sc@n@list\expandafter\sc@n@list\rem@inder|
1201 \ifx\rem@inder\empty\advance\l@tempcnta\@ne\fi
1202 \fi
1203 \ch@ck@l@ckR
1204 \fi
1205 \ifnum\l@tempcnta=\l@tempcntb
1206 \ifl@skipversenumberR\else
1207 \iftwocolumn

```

```

1207 \if@firstcolumn
1208 \gdef\l@dld@ta{\llap{{\leftlinenumR}}}%
1209 \else
1210 \gdef\l@drd@ta{\rlap{{\rightlinenumR}}}%
1211 \fi
1212 \else
1213 \l@dttempcntb=\line@marginR
1214 \ifnum\l@dttempcntb>\@ne
1215 \advance\l@dttempcntb by\page@numR
1216 \fi
1217 \ifodd\l@dttempcntb
1218 \gdef\l@drd@ta{\rlap{{\rightlinenumR}}}%
1219 \else
1220 \gdef\l@dld@ta{\llap{{\leftlinenumR}}}%
1221 \fi
1222 \fi
1223 \fi
1224 \fi
1225 \f@x@l@cksR
1226 \fi
1227 \fi
1228 \fi}
1229 %

```

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@numL30
\affixpstart@numR31 \newcommand*{\affixpstart@numL}{%
\leftpstartnumR32 \ifsidepstartnum
\rightpstartnumR33 \if@twocolumn
\leftpstartnumL34 \if@firstcolumn
\rightpstartnumL35 \gdef\l@dld@ta{\llap{{\leftpstartnumL}}}%
\leftpstartnumR36 \else
\rightpstartnumL37 \gdef\l@drd@ta{\rlap{{\rightpstartnumL}}}%
\rightpstartnumR38 \fi
\leftpstartnumL39 \else
\rightpstartnumR40 \l@dttempcntb=\line@margin
\leftpstartnumL41 \ifnum\l@dttempcntb>\@ne
\rightpstartnumR42 \advance\l@dttempcntb \page@num
\leftpstartnumL43 \fi
\rightpstartnumR44 \ifodd\l@dttempcntb

```



```

1245     \gdef\l@drd@ta{\rlap{{\rightpstartnumL}}}%
1246     \else
1247     \gdef\l@dld@ta{\llap{{\leftpstartnumL}}}%
1248     \fi
1249     \fi
1250 \fi
1251 }
1252 \newcommand*{\affixpstart@numR}{%
1253 \ifsidepstartnum
1254 \if@twocolumn
1255     \if@firstcolumn
1256     \gdef\l@dld@ta{\llap{{\leftpstartnumR}}}%
1257     \else
1258     \gdef\l@drd@ta{\rlap{{\rightpstartnumR}}}%
1259     \fi
1260     \else
1261     \l@dttempcntb=\line@marginR
1262     \ifnum\l@dttempcntb>\@ne
1263     \advance\l@dttempcntb \page@numR
1264     \fi
1265     \ifodd\l@dttempcntb
1266     \gdef\l@drd@ta{\rlap{{\rightpstartnumR}}}%
1267     \else
1268     \gdef\l@dld@ta{\llap{{\leftpstartnumR}}}%
1269     \fi
1270     \fi
1271 \fi
1272 }
1273
1274 \newcommand*{\leftpstartnumL}{
1275 \ifpstartnum
1276 \thepstartL
1277 \kern\linenumsep\global\pstartnumfalse\fi
1278 }
1279 \newcommand*{\rightpstartnumL}{
1280 \ifpstartnum\kern\linenumsep
1281 \thepstartL
1282 \global\pstartnumfalse\fi
1283 }
1284 \newif\ifpstartnumR
1285 \pstartnumRtrue
1286 \newcommand*{\leftpstartnumR}{
1287 \ifpstartnumR
1288 \thepstartR
1289 \kern\linenumsep\global\pstartnumRfalse\fi
1290 }
1291 \newcommand*{\rightpstartnumR}{
1292 \ifpstartnumR\kern\linenumsep
1293 \thepstartR
1294 \global\pstartnumRfalse\fi

```

```
1295 }
1296 %
```

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.

```
1297 \list@create{\inserts@listR}
1298 %
```

`\add@insertsR` The right text version.
`\add@inserts@nextR`

```
1299 \newcommand*{\add@insertsR}{%
1300   \global\let\add@inserts@nextR=\relax
1301   \ifx\inserts@listR\empty \else
1302     \ifx\next@insertR\empty
1303       \ifx\insertlines@listR\empty
1304         \global\noteschanged@true
1305         \gdef\next@insertR{100000}%
1306       \else
1307         \gl@p\insertlines@listR\to\next@insertR
1308       \fi
1309     \fi
1310     \ifnum\next@insertR=\absline@numR
1311       \gl@p\inserts@listR\to\@insertR
1312       \@insertR
1313       \global\let\@insertR=\undefined
1314       \global\let\next@insertR=\empty
1315       \global\let\add@inserts@nextR=\add@insertsR
1316     \fi
1317   \fi
1318   \add@inserts@nextR}
1319 %
1320 %
```

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 paragraph; these penalties get removed by the `\vsplit` operation. `\displaywidowpenalty` and `\brokenpenalty` are not restored, since we have no easy way to find out where we should insert them.

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

been worked out in `\do@ballast`. Finally, the penalty is checked to see that it does not go below -10000 .

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

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

```
1321 \newcommand*{\add@penaltiesL}{}
1322 \newcommand*{\add@penaltiesR}{}
1323
1324 %
```

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.

```
1325 \newcommand*{\flush@notesR}{%
1326 \xloop
1327 \ifx\inserts@listR\empty \else
1328 \gl@p\inserts@listR\to\@insertR
1329 \@insertR
1330 \global\let\@insertR=\undefined
```

```

1331 \repeat}
1332
1333 %

```

IX Footnotes

IX.1 Line number printing

The `\printlinesR` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on V.9 p. 81 of Eledmac’ handbook: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

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

```

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

```

```

1334 \def\printlinesR#1|#2|#3|#4|#5|#6|#7|{\begingroup
1335   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
1336   \ifl@d@pnum #1\fullstop\fi
1337   \ifledplinenum \linenumr@p{#2}\@Rlineflag\else \symlinenum\fi
1338   \ifl@d@ssub \fullstop \sublinenumr@p{#3}\fi
1339   \ifl@d@dash \endashchar\fi
1340   \ifl@d@pnum #4\fullstop\fi
1341   \ifl@d@elin \linenumr@p{#5}\@Rlineflag\fi
1342   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumr@p{#6}\fi
1343 \endgroup}
1344
1345
1346 %

```

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 `\print@Xnotes@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed³:

- 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.

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

- 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.

```
1347 \newcommand\print@Xnotes@forpages[1]{%
1348 %
```

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

```

1349 \ifcseempty{Xonlyside@#1}{%
1350 \csuse{#1footstart}{#1}%
1351 \csuse{#1footgroup}{#1}%
1352 }%
1353 %

```

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

```

1354 {%
1355 \ifboolexpr{%
1356 ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\ifnumodd{\c@page
1357 }})%
1358 or%
1359 (test {\ifcsstring{Xonlyside@#1}{R}} and test{\ifnumodd{\c@page}}))%
1360 }%

```

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

```

1361 {%
1362 \correct@Xfootins@box{#1}%
1363 \csuse{#1footstart}{#1}%
1364 \csuse{#1footgroup}{#1}%
1365 %

```

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

```

1366 \global\count\csuse{#1footins}=0%
1367 \global\skip\csuse{#1footins}=0pt%
1368 %

```

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

```

1369 \csuse{Xnotefontsize@#1}%
1370 \global\advance\dimen\csuse{#1footins} by -\baselineskip%
1371 %

```

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

```

1372 \global\boolfalse{keepforXside@#1}%
1373 }%
1374 %

```

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.

```

1375 {%
1376 \global\booltrue{keepforXside@#1}%
1377 %

```

Then restore expected rooms for notes on the next page.

```

1378 \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1379 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
1380 %

```

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

```

1381 \bgroup%
1382 \csuse{Xnotefontsize@#1}%
1383 \global\advance\dimen\csuse{#1footins} by \baselineskip%
1384 \egroup%
1385 %

```

```

1386 % End of \protect\cs{print@Xnotes@forpages}.
1387 }%
1388 }%
1389 }%
1390 %

```

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

```

1391 \newcommand{\correct@Xfootins@box}[1]{%
1392 %

```

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.

```

1393 \ifbool{keepforXside@#1}{%
1394 %

```

Some setting needed to do the right splitting.

```

1395 \csuse{Xnotefontsize@#1}%
1396 \splittopskip=0pt%
1397 %

```

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

```

1398 \global\setbox\csuse{#1footins}=\vbox{%
1399 \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1400 \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1401 \unvbox\csuse{#1footins}%
1402 }%
1403 %

```

End of the macro.

```

1404 }{}%
1405 }%
1406 %

```

And now, the same for familiar footnotes.

```

1407 \newcommand\print@notesX@forpages[1]{%
1408   \ifcempty{onlysideX@#1}{%
1409     \csuse{footstart#1}{#1}%
1410     \csuse{footgroup#1}{#1}%
1411   }%
1412   {%
1413     \ifboolexpr{%
1414       ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page
1415       }%
1416       or%
1417       (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}}))%
1418     }%
1419     \correct@footinsX@box{#1}%
1420     \csuse{footstart#1}{#1}%
1421     \csuse{footgroup#1}{#1}%
1422     \global\count\csuse{footins#1}=0%
1423     \global\skip\csuse{footins#1}=0pt%
1424     \csuse{notefontsizeX@#1}%
1425     \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1426     \global\boolfalse{keepforsideX@#1}%
1427   }%
1428   {%
1429     \global\booltrue{keepforsideX@#1}%
1430     \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1431     \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1432     \bgroup%
1433       \csuse{notefontsizeX@#1}%
1434       \global\advance\dimen\csuse{footins#1} by \baselineskip%
1435     \egroup%
1436   }%
1437 }%
1438 }%
1439 \newcommand{\correct@footinsX@box}[1]{%
1440   \ifbool{keepforsideX@#1}{%
1441     \csuse{notefontsizeX@#1}%
1442     \splittopskip=0pt%
1443     \global\setbox\csuse{footins#1}=\vbox{%
1444       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1445       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1446       \unvbox\csuse{footins#1}%
1447     }%
1448   }{}%
1449 }%
1450 %

```


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.

```
1451 \list@create{\labelref@listR}
1452
1453 %
```

`\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`.

```
1454 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1455   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1456     \led@warn@DuplicateLabel{#4}%
1457   \fi
1458   \expandafter\gdef\csname the@label#5\endcsname{#1|#2\@Rlineflag|#3|#4}%
1459   \ignorespaces}
1460 \AtBeginDocument{%
1461   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1462   }
1463
1464 %
```

`\@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*
1465 \WithSuffix\newcommand\sidenotemargin*[1]{%
1466   \l@dgetsidenote@margin{#1}
1467   \global\sidenote@marginR=\@l@dttempcntb
1468   \global\sidenote@margin=\@l@dttempcntb
1469 }
1470 \newcount\sidenote@marginR
1471 \global\sidenote@margin=\@ne
1472
1473 %
```

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

```

1474 \newcommand*{\affixside@noteR}{%
1475   \def\sidenotecontent@{%
1476     \numgdef{\itemcount@}{0}%
1477     \def\do##1{%
1478       \ifnumequal{\itemcount@}{0}%
1479         {%
1480           \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
1481         {\appto\sidenotecontent@{\sidenotessep ##1}%
1482         }%
1483         \numgdef{\itemcount@}{\itemcount@+1}%
1484       }%
1485       \dolistloop{\l@dcstotetext}%
1486       \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1487       \gdef\@templ@d{%
1488         \gdef\@templ@n{\l@dcstotetext\l@dcstotetext@l\l@dcstotetext@r}%
1489         \ifx\@templ@d\@templ@n \else%
1490           \if@twocolumn%
1491             \if@firstcolumn%
1492               \setl@dlp@rbox{##1}{\sidenotecontent@}%
1493             \else%
1494               \setl@drp@rbox{\sidenotecontent@}%
1495             \fi%
1496           \else%
1497             \l@dttempcntb=\sidenote@marginR%
1498             \ifnum\l@dttempcntb>\@ne%
1499               \advance\l@dttempcntb by\page@numR%
1500             \fi%
1501             \ifodd\l@dttempcntb%
1502               \setl@drp@rbox{\sidenotecontent@}%
1503             \gdef\sidenotecontent@{%
1504               \numdef{\itemcount@}{0}%
1505               \dolistloop{\l@dcstotetext@l}%
1506               \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1507               \setl@dlp@rbox{\sidenotecontent@}%
1508             \else%
1509               \setl@dlp@rbox{\sidenotecontent@}%
1510             \gdef\sidenotecontent@{%
1511               \numdef{\itemcount@}{0}%
1512               \dolistloop{\l@dcstotetext@r}%
1513               \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1514               \setl@drp@rbox{\sidenotecontent@}%
1515             \fi%
1516           \fi%
1517         \fi%
1518       }
1519     }%
1520   }%

```

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.

```

\inserthangingsymbolL21 \newif\ifinserthangingsymbolR
\inserthangingsymbolR22 \newcommand{\inserthangingsymbolL}{%
1523   \ifinserthangingsymbol%
1524   \ifinstanzaL%
1525       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1526         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1527   \hangingsymbol%
1528   \fi%
1529 \fi%
1530 }%
1531 \newcommand{\inserthangingsymbolR}{%
1532   \ifinserthangingsymbolR%
1533   \ifinstanzaR%
1534       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1535         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1536   \hangingsymbol%
1537   \fi%
1538 \fi%
1539 }%
1540 %

```

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.

```

1541 \chardef\next=\catcode`\&
1542 \catcode`\&=\active
1543
1544 %

```

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

```

1545 \newenvironment{astanza}[1][\&]{%
1546   \catcode`\&=\active
1547   \global\stanza@count\@ne\stanza@modulo\@ne

```

```

1548 \ifnum\usernamecount{sza@0@}=\z@
1549 \let\stanza@hang\relax
1550 \let\endlock\relax
1551 \else
1552 \rightskip\z@ plus 1fil\relax
1553 \fi
1554 \ifnum\usernamecount{szp@0@}=\z@
1555 \let\sza@penalty\relax
1556 \fi
1557 \def&{%
1558 \endlock\mbox{}%
1559 \sza@penalty
1560 \global\advance\stanza@count\@ne
1561 \@astanza@line}%
1562 \def\&{\@stopastanza}%
1563 \pstart[#1]%
1564 \@astanza@line
1565 }{}
1566
1567 %

```

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

```

1568 \newcommandx{\@stopastanza}[1][1,usedefault]{%
1569 \endlock\mbox{}%
1570 \pend[#1]%
1571 }%
1572 %

```

\@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.

```

1573 \newcommand*{\@astanza@line}{%
1574 \ifnum\value{stanzaindentsrepetition}=0
1575 \parindent=\csname sza@\number\stanza@count
1576 @\endcsname\stanzaindentbase
1577 \else
1578 \parindent=\csname sza@\number\stanza@modulo
1579 @\endcsname\stanzaindentbase
1580 \managestanza@modulo
1581 \fi
1582 \par
1583 \stanza@hang%\mbox{}%
1584 \ignorespaces}
1585
1586 %

```

Lastly reset the modified category codes.

```

1587 \catcode`\&=\next
1588
1589 %

```

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

```

\thestanzaR
1590 \newcounter{stanzaL}
1591 \newcounter{stanzaR}
1592 \renewcommand{\thestanzaL}{%
1593   \textbf{\arabic{stanzaL}}%
1594 }
1595 \renewcommand{\thestanzaR}{%
1596   \textbf{\arabic{stanzaR}}%
1597 }
1598 %
1599 %

```

XIV Naming macros

The \TeX kernel provides `\@namedef` and `\@namuse` for defining and using macros that may have non-letters in their names. We need something similar here as we are going to need and use some numbered boxes and counters.

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

```

\setnamebox
\unhnamebox
\unvnamebox
\namebox
1600 \providecommand*\newnamebox}[1]{%
1601   \expandafter\newbox\csname #1\endcsname}
1602 \providecommand*\setnamebox}[1]{%
1603   \expandafter\setbox\csname #1\endcsname}
1604 \providecommand*\unhnamebox}[1]{%
1605   \expandafter\unhbox\csname #1\endcsname}
1606 \providecommand*\unvnamebox}[1]{%
1607   \expandafter\unvbox\csname #1\endcsname}
1608 \providecommand*\namebox}[1]{%
1609   \csname #1\endcsname}
1610
1611 %

```

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

```

\usenamecount
1612 \providecommand*\newnamecount}[1]{%
1613   \expandafter\newcount\csname #1\endcsname}
1614 \providecommand*\usenamecount}[1]{%
1615   \csname #1\endcsname}
1616
1617 %

```

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@dusedbabeltrue1618 \newif\ifl@dusedbabel
1619 %
```

```
\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 \.
```

```
1620 \patchcmd{\bbl@set@language}%
1621 {\select@language{\language}}}%
1622 {\edef\language{#1}\select@language{\language}}}%
1623 {}%
1624 {}%
1625
1626 %
```

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
1627 \newcommand*{\l@duselanguage}[1]{%
1628 \gdef\theledlanguageL{#1}
1629 \gdef\theledlanguageR{#1}
1630
1631 %
```

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

```

1632 \AtBeginDocument{%
1633   \ifundefined{xpg@main@language}{%
1634     \ifundefined{bbl@main@language}{%
1635 %

```

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

```

1636   \l@dusedbabelfalse
1637 }{%
1638 %

```

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.

```

1639   \l@dusedbabeltrue
1640   \let\l@doldselectlanguage\selectlanguage
1641   \let\l@doldbbl@set@language\bbl@set@language
1642   \renewcommand{\selectlanguage}[1]{%
1643     \l@doldselectlanguage{#1}%
1644     \ifledRcol \gdef\theledlanguageR{#1}%
1645     \else      \gdef\theledlanguageL{#1}%
1646     \fi}
1647 %

```

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

```

1648   \renewcommand*{\l@duselanguage}[1]{%
1649     \l@doldselectlanguage{#1}}
1650 %

```

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

```

1651   \gdef\theledlanguageL{\bbl@main@language}%
1652   \gdef\theledlanguageR{\bbl@main@language}%
1653   }%
1654 }
1655 %

```

If use polyglossia

```

1656 { \let\old@otherlanguage\otherlanguage%
1657   \renewcommand{\otherlanguage}[2] []{%
1658     \selectlanguage{#1}{#2}%
1659     \ifledRcol \gdef\theledlanguageR{#2}%
1660     \else      \gdef\theledlanguageL{#2}%
1661     \fi}%
1662   \let\l@duselanguage\select@language%
1663   \gdef\theledlanguageL{\xpg@main@language}%
1664   \gdef\theledlanguageR{\xpg@main@language}%
1665 %

```

That is it.

```
1666 }}
1667 %
```

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.
`\l@dc@maxchunks`

```
1668 \newcount\l@dc@maxchunks
1669 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1670 \maxchunks{5120}
1671
1672 %
```

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

```
1673 \newcount\l@dnumpstartsR
1674
1675 %
```

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

`\l@pscr`

```
1676 \newcount\l@dpscL
1677 \newcount\l@dpscR
1678
1679 %
```

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

```
1680 \newcommand*\l@dsetuprawboxes{%
1681   \@l@tempcntb=\l@dc@maxchunks
1682   \loop\ifnum\@l@tempcntb>\z@
1683     \newnamebox{\l@dLcolrawbox\the\@l@tempcntb}
1684     \newnamebox{\l@dRcolrawbox\the\@l@tempcntb}
1685     \advance\@l@tempcntb \m@ne
1686   \repeat}
1687
1688 %
```

`\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.


```

1689 \newcommand*{\l@dsetupmaxlinecounts}{%
1690   \l@dtempcntb=\l@dc@maxchunks
1691   \loop\ifnum\l@dtempcntb>\z@
1692     \newnamecount{\l@dmxlinesinpar\the\l@dtempcntb}
1693     \advance\l@dtempcntb \m@ne
1694   \repeat}
1695 \newcommand*{\l@dzeromaxlinecounts}{%
1696   \begingroup
1697   \l@dtempcntb=\l@dc@maxchunks
1698   \loop\ifnum\l@dtempcntb>\z@
1699     \global\usenamecount{\l@dmxlinesinpar\the\l@dtempcntb}=\z@
1700     \advance\l@dtempcntb \m@ne
1701   \repeat
1702   \endgroup}
1703
1704 %

```

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

```

1705 \AtBeginDocument{%
1706   \l@dsetuprawboxes
1707   \l@dsetupmaxlinecounts
1708   \l@dzeromaxlinecounts
1709   \l@dnumpsstartsL=\z@
1710   \l@dnumpsstartsR=\z@
1711   \l@dpscL=\z@
1712   \l@dpscR=\z@}
1713
1714 %

```

XVII Checking text to be processed

```

\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue
\@pstartsfalse
\check@pstarts
1715 \newif\if@pstarts
1716 \newcommand*{\check@pstarts}{%
1717   \@pstartsfalse
1718   \ifnum\l@dnumpsstartsL>\l@dpscL
1719     \@pstartstrue
1720   \else
1721     \ifnum\l@dnumpsstartsR>\l@dpscR
1722       \@pstartstrue
1723     \fi
1724   \fi
1725 }
1726
1727 %

```

`\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`.
`\araw@texttrue`
`\araw@textfalse`
`\checkraw@text`

```

1728 \newif\ifaraw@text
1729 \newcommand*\checkraw@text{%
1730   \araw@textfalse
1731   \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}
1732     \araw@texttrue
1733   \else
1734     \ifvbox\namebox{1@dRcolrawbox\the\1@dpscR}
1735       \araw@texttrue
1736     \fi
1737   \fi
1738 }
1739
1740 %

```

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

```

1741 \newcommand*\@writelinesinparL{%
1742   \edef\next{%
1743     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1744   \next
1745   \global\@donereallinesL \z@
1746 \newcommand*\@writelinesinparR{%
1747   \edef\next{%
1748     \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1749   \next
1750   \global\@donereallinesR \z@
1751
1752 %

```

XVIII Parallel columns

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

`\@eledsectionR`

```

1753 \newsavebox{\@eledsectionL}%
1754 \newsavebox{\@eledsectionR}%
1755 %

```

`\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.

```

1756 \newcommand*\Columns{%
1757   \ifl@dpairing%
1758     \led@err@Columns@InsideEnv%
1759   \fi%

```

```

1760 \l@dprintingcolumnstrue%
1761 \eledsection@correcting@skip=-\baselineskip% Correction for sections'
titles
1762 \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1763 \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1764 \fi
1765 %

```

Start a group and zero counters, etc.

```

1766 \begingroup
1767 \l@dzeropenalties
1768 \endgraf\global\num@lines=\prevgraf
1769 \global\num@linesR=\prevgraf
1770 \global\par@line=\z@
1771 \global\par@lineR=\z@
1772 \global\l@dpscL=\z@
1773 \global\l@dpscR=\z@
1774 %

```

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

```

1775 \check@pstarts
1776 \loop\if@pstarts
1777 \global\pstartnumtrue
1778 \global\pstartnumRtrue
1779 %

```

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.

```

1780 \global\advance\l@dpscL \@ne
1781 \global\advance\l@dpscR \@ne
1782 \restore@pstartL@pc%
1783 \restore@pstartR@pc%
1784 %

```

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

```

1785 \Columns@print@before@pstart%
1786 %

```

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

```

1787 \checkraw@text
1788 { \loop\ifaraw@text
1789 %

```

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

```

1790 \l@duselanguage{\theledlanguageL}%
1791 \do@lineL

```

```

1792 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1793 {%
1794 \ifdefstring{\@eledsectmark}{L}%
1795 {\csuse{eled@sectmark@the\l@dpscL}%
1796 }{}%
1797 \global\csundef{eled@sectmark@the\l@dpscL}%
1798 \savebox{\@eledsectionL}{\parbox[t]{}[t]{\Lcolwidth}{\vbox
{} \print@eledsectionL}}% \vbox{}-> prevent alignment troubles with RTL
language
1799 }%
1800 {}%
1801 \l@duselanguage{\theledlanguageR}%
1802 \do@lineR
1803 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1804 {%
1805 \ifdefstring{\@eledsectmark}{R}%
1806 {\csuse{eled@sectmark@the\l@dpscR}%
1807 }{}%
1808 \global\csundef{eled@sectmark@the\l@dpscR}%
1809 \savebox{\@eledsectionR}{\parbox[t]{}[t]{\Rcolwidth}{\vbox
{} \print@eledsectionR}}% \vbox{}-> prevent alignment troubles with RTL
language
1810 {}%
1811 \hb@xt@ \hsize{%
1812 \ifdefstring{\columns@position}{L}{\hfill}%
1813 \unhbox\l@leftbox%
1814 \ifhbox\@eledsectionL%
1815 \usebox{\@eledsectionL}%
1816 \fi%
1817 \print@columnseparator%
1818 \unhbox\l@rightbox%
1819 \ifhbox\@eledsectionR%
1820 \usebox{\@eledsectionR}%
1821 \fi%
1822 \ifdefstring{\columns@position}{R}{\hfill}%
1823 }%
1824 \checkraw@text
1825 \checkverseL
1826 \checkverseR
1827 \checkpb@columns
1828 \repeat}
1829 %

```

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`.

```

1830 \@writelinesinparL
1831 \@writelinesinparR
1832 \check@pstarts
1833 \ifbypstart@%

```

```

1834         \write\linenum@out{\string\@set[1]}
1835         \resetprevline@
1836     \fi
1837     \ifbypstart@R
1838         \write\linenum@outR{\string\@set[1]}
1839         \resetprevline@
1840     \fi
1841     \Columns@print@after@pend%
1842 \repeat
1843 %

```

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.

```

1844     \flush@notes
1845     \flush@notesR
1846 \endgroup
1847 %

```

```

1848 \global\l@dpscL=\z@
1849 \global\l@dpscR=\z@
1850 \global\l@dnumstartsL=\z@
1851 \global\l@dnumstartsR=\z@
1852 \l@dprintingcolumnsfalse%
1853 \ignorespaces
1854 \global\instanzaLfalse
1855 \global\instanzaRfalse}
1856
1857 %

```

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

```

1858 \def\print@columnseparator{%
1859     \ifdim\beforecolumnseparator<0pt%
1860         \hfill%
1861     \else%
1862         \hspace{\beforecolumnseparator}%
1863     \fi%
1864     \columnseparator%
1865     \ifdim\aftercolumnseparator<0pt%
1866         \hfill%
1867     \else%
1868         \hspace{\aftercolumnseparator}%
1869     \fi%
1870 }%
1871 %

```

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

```

1872
1873 \newcommand{\checkpb@columns}{%
1874     \newif\if@pb
1875     \newif\if@nopb
1876     \IfStrEq{\led@pb@setting}{before}{
1877         \numdef{\next@absline}{\the\absline@num+1}%
1878         \numdef{\next@abslineR}{\the\absline@numR+1}%
1879         \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
1880         \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
1881         \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
1882         \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
1883         }{}
1884         \IfStrEq{\led@pb@setting}{after}{
1885         \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
1886         \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
1887         \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
1888         \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
1889         }{}
1890         \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
1891         \if@pb\pagebreak[4]\fi
1892     }
1893 %

```

\columnseparator The separator between line pairs in parallel columns is in the form of a vertical rule extending a little below the baseline and with a height slightly greater than the **\baselineskip**. The width of the rule is **\columnrulewidth** (initially 0pt so the rule is invisible).

```

1894 \newcommand*{\columnseparator}{%
1895     \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}
1896     \newdimen\columnrulewidth
1897     \columnrulewidth=\z@
1898
1899 %

```

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

```

1900 \newcommand*{\columnspan}[1]{%
1901     \xdef\columns@position{#1}%
1902 }%
1903 \xdef\columns@position{R}%
1904 %

```

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

```

1905 \newlength{\beforecolumnseparator}%
1906 \setlength{\beforecolumnseparator}{-2pt}%

```

```

1907
1908 \newlength{\aftercolumnseparator}%
1909 \setlength{\aftercolumnseparator}{-2pt}%
1910
1911 %

```

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

```

1912 \newcommand{\setwidthliketwocolumns@L}{%
1913 % Temporary dimension, initially equal to the standard hsize, i.e. text
1914 % width
1915 % \begin{macrocode}
1916 \newdimen\temp%
1917 \temp=\hsize%
1918 %

```

Hsize : Left + Right width

```

1918 \hsize=\Lcolwidth%
1919 \advance\hsize\Rcolwidth%
1920 %

```

Now, calculating the remaining space

```

1921 \advance\temp-\hsize%
1922 %

```

And multiply the hsize by 2/3 of this space

```

1923 \multiply\temp by 2%
1924 \divide\temp by 3%
1925 \advance\hsize\temp%
1926 }%
1927
1928 \newcommand{\setpositionliketwocolumns@L}{%
1929 \renewcommand{\ledrlfill}{\hfill}%
1930 }%
1931
1932 \newcommand{\setnotespositionliketwocolumns@L}{%
1933 }%
1934
1935 %
1936 %

```

```

1937 \newcommand{\setwidthliketwocolumns@C}{%
1938 % Temporary dimension, initially equal to the standard hsize, i.e. text
1939 % width

```

```

1940 \newdimen\temp%
1941 \temp=\hsize%
1942 % Hsize : Left + Right width
1943 %

```

```

1944 \hsize=\Lcolwidth%
1945 \advance\hsize\Rcolwidth%
1946 % Now, calculating the remaining space
1947 %

```

```

1948 \advance\temp-\hsize%
1949 %

```

And multiply the hsize by 1/2 of this space

```

1950 \divide\temp by 2%
1951 \advance\hsize\temp%
1952 }%
1953
1954 \newcommand{\setpositionliketwocolumns@C}{%
1955 \doinsidelinehook{\hfill}%
1956 \renewcommand{\ledrlfill}{\hfill}%
1957 }%
1958
1959 \newcommand{\setnotespositionliketwocolumns@C}{%
1960 \newdimen\temp%
1961 \newdimen\tempa%
1962 \temp=\hsize%
1963 \tempa=\Lcolwidth%
1964 \advance\tempa\Rcolwidth%
1965 \advance\temp-\tempa%
1966 \divide\temp by 2%
1967 \leftskip=\temp%
1968 \rightskip=-\temp%
1969 }%
1970
1971 \newcommand{\setwidthliketwocolumns@R}{%
1972 %

```

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

```

1973 \newdimen\temp%
1974 \temp=\hsize%
1975 %

```

Hsize : Left + Right width

```

1976 \hsize=\Lcolwidth%
1977 \advance\hsize\Rcolwidth%
1978 %

```

Now, calculating the remaining space


```

1979 \advance\temp-\hsize%
1980 %
1981
1982 And multiply the hsize by 2/3 of this space
1983
1984 \multiply\temp by 2%
1985 \divide\temp by 3%
1986 \advance\hsize\temp%
1987 }%
1988
1989 \newcommand{\setpositionliketwocolumns@R}{%
1990 \doinsidelinehook{\hfill}%
1991 }%
1992
1993 \newcommand{\setnotespositionliketwocolumns@R}{%
1994 \newdimen\temp%
1995 \newdimen\tempa%
1996 \temp=\hsize%
1997 \tempa=\Lcolwidth%
1998 \advance\tempa\Rcolwidth%
1999 \advance\temp-\tempa%
2000 \divide\temp by 2%
2001 \leftskip=\temp%
2002 \rightskip=-\temp%
2003 }%
2004 %
2005

```

`\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.

```

2003 \newcommand{\Columns@print@before@pstart}{%
2004 \ifboolexpr{%
2005 test{\ifcsstring{before@pstartL@the\l@dpscL}{\at@every@pstart}}%
2006 and test {\ifcsstring{before@pstartR@the\l@dpscR}{\at@every@pstart}}%
2007 and test {\ifdefempty{\at@every@pstart}}}%
2008 }{%
2009 {%
2010 \hb@xt@ \hsize{%
2011 \ifdefstring{\columns@position}{L}{\hfill }%
2012 \par\parbox[t]{}[t]{\Lcolwidth}{%
2013 \csuse{before@pstartL@the\l@dpscL}%
2014 }%
2015 \print@columnseparator%
2016 \parbox[t]{}[t]{\Rcolwidth}{%
2017 \csuse{before@pstartR@the\l@dpscR}%
2018 }%
2019 \ifdefstring{\columns@position}{R}{\hfill}%
2020 }%
2021 }%

```

```

2022 \global\csundef{before@pstartL@the\l@dpscL}%
2023 \global\csundef{before@pstartR@the\l@dpscR}%
2024 }%
2025 \newcommand{\Columns@print@after@pend}{%
2026   \ifboolexpr{%
2027     test{\ifcsstring{after@pendL@the\l@dpscL}{\at@every@pend}}%
2028     and test {\ifcsstring{after@pendR@the\l@dpscR}{\at@every@pend}}%
2029     and test {\ifdefempty{\at@every@pend}}}%
2030     {%
2031     {%
2032       \hb@xt@ \hsize{%
2033         \ifdefstring{\columns@position}{L}{\hfill }%
2034         \parbox[t] [] [t]{\Lcolwidth}{%
2035           \csuse{after@pendL@the\l@dpscL}%
2036         }%
2037         \print@columnseparator%
2038         \parbox[t] [] [t]{\Rcolwidth}{%
2039           \csuse{after@pendR@the\l@dpscR}%
2040         }%
2041         \ifdefstring{\columns@position}{R}{\hfill}%
2042       }%
2043     }%
2044     \global\csundef{after@pendL@the\l@dpscL}%
2045     \global\csundef{after@pendR@the\l@dpscR}%
2046   }%
2047   %

```

XIX Parallel pages

This is considerably more complicated than parallel columns.

XIX.1 Specific counters

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

```

2048 \newcount\numpagelinesL
2049 \newcount\numpagelinesR
2050 \newcount\l@dminpagelines
2051
2052 %

```

XIX.2 Main macro

<code>\Pages</code>	The <code>\Pages</code> 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.
---------------------	---

```

2053 \newcommand*{\Pages}{%
2054   \ifl@dpairing%
2055     \led@err@Pages@InsideEnv%
2056   \fi%
2057   \eledsection@correcting@skip=-2\baselineskip% line correcting for section
titles.
2058   \parledgroup@notespacing@set@correction%
2059   \typeout{}%
2060   \typeout{***** PAGES *****}%
2061   \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else%
2062     \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2063   \fi%
2064 %

```

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

```

2065 \setlength{\Lcolwidth}{\textwidth}%
2066 \setlength{\Rcolwidth}{\textwidth}%
2067 %

```

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

```

2068 \cleartol@evenpage%
2069 \l@dprintingpagetrue%
2070 \begingroup%
2071   \l@dzeropenalties%
2072   \endgraf\global\num@lines=\prevgraf%
2073   \global\num@linesR=\prevgraf%
2074   \global\par@line=\z@%
2075   \global\par@lineR=\z@%
2076   \global\l@dpscL=\z@%
2077   \global\l@dpscR=\z@%
2078   \writtenlinesLfalse%
2079   \writtenlinesRfalse%
2080 %

```

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

```

2081 \let\print@Xnotes\print@Xnotes@forpages%
2082 \let\print@notesX\print@notesX@forpages%
2083 %

```

Check if there are chunks to be processed.

```

2084 \check@pstarts%
2085 \loop\if@pstarts%
2086 %

```

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

```

2087 \global\advance\l@dpscL \@ne%
2088 \global\advance\l@dpscR \@ne%
2089 %

```

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

```

2090 \getlinesfromparlistL%
2091 \getlinesfromparlistR%
2092 \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2093 {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
2094 \check@pstarts%
2095 \repeat%
2096 %

```

Zero the counts again, ready for the next bit.

```

2097 \global\l@dpscL=\z@%
2098 \global\l@dpscR=\z@%
2099 %

```

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

```

2100 \getlinesfrompagelistL%
2101 \getlinesfrompagelistR%
2102 \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2103 {\l@dminpagelines}%
2104 %

```

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.

```

2105 \check@pstarts%
2106 \if@pstarts%
2107 %

```

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

```

2108 \global\advance\l@dpscL \@ne%
2109 \global\advance\l@dpscR \@ne%
2110 \restore@pstartL@pc%
2111 \restore@pstartR@pc%
2112 %

```

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

```

2113 \global\@donereallinesL=\z@%
2114 \global\@donetotallinesL=\z@%
2115 \global\@donereallinesR=\z@%
2116 \global\@donetotallinesR=\z@%
2117 %

```

Start a loop over the boxes (chunks).

```

2118 \checkraw@text%
2119 %
2120 % \beginngroup
2121 { \loop\ifaraw@text%
2122 %

```

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

```

2123 \checkpageL%
2124 \l@duselanguage{\theledlanguageL}%
2125 { \loop\ifl@dsamepage%
2126 %

```

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

```

2127 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2128 \csuse{before@pstartL@the\l@dpscL}%
2129 \global\csundef{before@pstartL@the\l@dpscL}%
2130 \do@lineL%
2131 \xifinlist{\the\l@dpscL}{\eled@sections@@}
2132 {\print@eledsectionL}%
2133 {}%
2134 \advance\numpagelinesL \@ne%
2135 %

```

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

```

2136 \ifshiftedpstarts%
2137 \ifdim\ht\l@dleftbox>0pt%
2138 \parledgroup@correction@notespacing{L}%
2139 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2140 \else%
2141 \dimen0=\pagetotal%
2142 \advance\dimen0 by \baselineskip%
2143 \global\pagetotal=\dimen0%
2144 \fi%
2145 \else%
2146 \parledgroup@correction@notespacing{L}%
2147 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2148 \fi%
2149 %

```

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

```

2150         \get@nextboxL%
2151     \global\l@dskipversenumberfalse%
2152         \ifprint@last@after@pendL%
2153             \csuse{after@pendL@the\l@dpscL}%
2154         \global\csundef{after@pendL@the\l@dpscL}%
2155         \fi%
2156         \checkpageL%
2157         \checkverseL%
2158         \checkpbL%
2159     \repeat%
2160 %

```

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.

```

2161         \ifl@dpagfull%
2162             \@writelinesonpageL{\the\numpagelinesL}%
2163         \else%
2164             \@writelinesonpageL{1000}%
2165         \fi%
2166 %

```

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.

```

2167         \numpagelinesL \z@%
2168         \parledgroup@correction@notespacing@init%
2169         \clearl@dleftpage }%
2170 %

```

Now do the same for the right text.

```

2171         \checkpageR%
2172         \l@duselanguage{\theledlanguageR}%
2173     {
2174         \loop\ifl@dsamepage%
2175             \initnumbering@sectcountR%
2176             \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2177             \csuse{before@pstartR@the\l@dpscR}%
2178             \global\csundef{before@pstartR@the\l@dpscR}%
2179             \do@lineR%
2180             \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
2181             {\print@eledsectionR}%
2182             {}%
2183             \advance\numpagelinesR \@ne%
2184             \ifshiftedpstarts%
                \ifdim\ht\l@drightbox>0pt%

```

```

2185         \parledgroup@correction@notespacing{R}%
2186         \hb@xt@ \hsz{\ledstrutR\unhbox\l@drightbox}%
2187     \else%
2188         \dimen0=\pagetotal%
2189         \advance\dimen0 by \baselineskip%
2190         \global\pagetotal=\dimen0%
2191     \fi%
2192 \else%
2193     \parledgroup@correction@notespacing{R}%
2194     \hb@xt@ \hsz{\ledstrutR\unhbox\l@drightbox}%
2195 \fi%
2196 \get@nextboxR%
2197 \global\l@dskipversenumberRfalse%
2198     \ifprint@last@after@pendR%
2199         \csuse{after@pendR@the\l@dpscR}%
2200         \global\csundef{after@pendR@the\l@dpscR}%
2201     \fi%
2202     \checkpageR%
2203     \checkverseR%
2204     \checkpbR%
2205     \repeat%
2206     \ifl@dpagfull%
2207         \@writelinesonpageR{\the\numpagelinesR}%
2208     \else%
2209         \@writelinesonpageR{1000}%
2210     \fi%
2211     \numpagelinesR=\z@%
2212     \parledgroup@correction@notespacing@init%
2213 %

```

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

```

2214     \clearl@drightpage}%
2215 %

```

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.

```

2216     \checkraw@text%
2217     \ifaraw@text%
2218         \getlinesfrompagelistL%
2219         \getlinesfrompagelistR%
2220         \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2221         {\l@dminpagelines}%
2222     \fi%
2223     \repeat}%
2224 %

```

We have now output the text from all the chunks.

```

2225 \fi%
2226 %

```

Make sure that there are no inserts hanging around.

```
2227 \flush@notes%
2228 \flush@notesR%
2229 \endgroup%
2230 %
```

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

```
2231 \global\l@dpscL=\z@%
2232 \global\l@dpscR=\z@%
2233 \global\l@dnumstartsL=\z@%
2234 \global\l@dnumstartsR=\z@%
2235 \global\instanzaLfalse%
2236 \global\instanzaRfalse%
2237 \l@dprintingpagesfalse%
2238 \finish@Pages@notes%Needed to prevent final notes overlap line number
2239 \ignorespaces}
2240
2241
2242 %
```

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.

```
2243 \newcommand{\finish@Pages@notes}{%
2244 \def\do##1{%
2245 %
```

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

```
2246 \ifnocritical@%
2247 \global\newnamebox{##1footins}
2248 \fi
2249 \ifnofamiliar@%
2250 \global\newnamebox{footins##1}
2251 \fi
2252 %
```

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

```
2253 \ifvoid\csuse{##1footins}%
2254 \ifvoid\csuse{footins##1}\else%
2255 \newpage\null%
2256 \listbreak%
2257 \fi%
2258 \else%
2259 \newpage\null%
```



```

2260 \listbreak%
2261 \fi%
2262 }%
2263 \dolistloop{\@series}%
2264 }%
2265 %

```

XIX.4 Struts

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

```

\ledstrutR
2266 \newcommand*\ledstrutL{\strut}
2267 \newcommand*\ledstrutR{\strut}
2268
2269 %

```

XIX.5 Page clearing

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

```

2270 \providecommand{\cleartoevenpage}[1][\@empty]{%
2271 \clearpage
2272 \ifodd\c@page\hbox{ }#1\clearpage\fi}
2273
2274 \newcommand*\cleartol@devenpage{%
2275 \ifdim\pagetotal<\topskip% on an empty page
2276 \else
2277 \clearpage
2278 \fi
2279 \ifodd\c@page%
2280 \ifprevpgnotnumbered%
2281 \addtocounter{par@page}{-1}%
2282 \ifdef{\prevpgstyle}{\thispagestyle{\prevpgstyle}}}%
2283 \fi%
2284 \hbox{ }\clearpage%
2285 \fi%
2286 }%
2287 %

```

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

```

2288 \newcommand*{\clearl@leftpage}{%
2289   \ifdim\pagetotal=0pt\hbox{}\fi%
2290   \newpage%
2291   \ifodd\c@page\else
2292     \led@err@LeftOnRightPage
2293     \hbox{}\fi%
2294   \cleardoublepage
2295   \fi}
2296
2297 \newcommand*{\clearl@drighthouse}{%
2298   \ifdim\pagetotal=0pt\hbox{}\fi%
2299   \newpage%
2300   \ifodd\c@page
2301     \led@err@RightOnLeftPage
2302     \hbox{}\fi%
2303   \cleartoevenpage
2304   \fi}
2305
2306 %

```

XIX.6 Lines managing

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

`\@cs@linesinparR`

```

2307 \newcommand*{\getlinesfromparlistL}{%
2308   \ifx\linesinpar@listL\empty
2309     \gdef\@cs@linesinparL{0}%
2310   \else
2311     \gl@p\linesinpar@listL\to\@cs@linesinparL
2312   \fi}
2313 \newcommand*{\getlinesfromparlistR}{%
2314   \ifx\linesinpar@listR\empty
2315     \gdef\@cs@linesinparR{0}%
2316   \else
2317     \gl@p\linesinpar@listR\to\@cs@linesinparR
2318   \fi}
2319
2320 %

```

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

`\@cs@linesonpageR`

```

2321 \newcommand*{\getlinesfrompagelistL}{%
2322   \ifx\linesonpage@listL\empty
2323     \gdef\@cs@linesonpageL{1000}%
2324   \else

```

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

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

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

XIX.7 Page break managing

<code>\ifl@dsamepage</code>	<code>\checkpageL</code> tests if the space and lines already taken on the page by text and foot-
<code>\l@dsamepagetrue</code>	notes is less than the constraints. If so, then <code>\ifl@dpagefull</code> is set FALSE and
<code>\l@dsamepagefalse</code>	<code>\ifl@dsamepage</code> is set TRUE. If the page is spatially full then <code>\ifl@dpagefull</code> is set
<code>\ifl@dpagefull</code>	TRUE and <code>\ifl@dsamepage</code> is set FALSE. If it is not spatially full but the maximum
<code>\l@dpagefulltrue</code>	
<code>\l@dpagefullfalse</code>	
<code>\checkpageL</code>	
<code>\checkpageR</code>	

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

```

2357 \newif\ifl@dsamepage
2358   \l@dsamepagetrue
2359 \newif\ifl@dpagfull
2360
2361 \newcommand*{\checkpageL}{%
2362   \l@dpagfulltrue
2363   \l@dsamepagetrue
2364   \check@goal
2365   \ifdim\pagetotal<\ledthegoal
2366     \ifnum\numpagelinesL<\l@dminpagelines
2367       \else
2368         \l@dsamepagefalse
2369         \l@dpagfullfalse
2370       \fi
2371     \else
2372       \l@dsamepagefalse
2373       \l@dpagfulltrue
2374     \fi%
2375   \ifprint@last@after@pendL%
2376     \l@dpagfullfalse%
2377     \l@dsamepagefalse%
2378     \print@last@after@pendLfalse%
2379   \fi%
2380 }%
2381
2382 \newcommand*{\checkpageR}{%
2383   \l@dpagfulltrue
2384   \l@dsamepagetrue
2385   \check@goal
2386   \ifdim\pagetotal<\ledthegoal
2387     \ifnum\numpagelinesR<\l@dminpagelines
2388       \else
2389         \l@dsamepagefalse
2390         \l@dpagfullfalse
2391       \fi
2392     \else
2393       \l@dsamepagefalse
2394       \l@dpagfulltrue
2395     \fi%
2396   \ifprint@last@after@pendR%
2397     \l@dpagfullfalse%
2398     \l@dsamepagefalse%
2399     \print@last@after@pendRfalse%
2400   \fi%
2401 }%
2402
2403 %

```

`\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`.

```

2404 \newcommand{\checkpbL}{
2405   \IfStrEq{\led@pb@setting}{after}{
2406     \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@dpagfulltrue\
l@dsamepagefalse}{
2407     \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@dpagfullfalse\
l@dsamepagetrue}{
2408     }{
2409     \IfStrEq{\led@pb@setting}{before}{
2410       \numdef{\next@absline}{\the\absline@num+1}
2411       \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagfulltrue\
l@dsamepagefalse}{
2412       \xifinlistcs{\next@absline}{l@prev@nopb}{\l@dpagfullfalse\
l@dsamepagetrue}{
2413       }{
2414     }
2415   }
2416 \newcommand{\checkpbR}{
2417   \IfStrEq{\led@pb@setting}{after}{
2418     \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@dpagfulltrue\
l@dsamepagefalse}{
2419     \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@dpagfullfalse\
l@dsamepagetrue}{
2420     }{
2421     \IfStrEq{\led@pb@setting}{before}{
2422       \numdef{\next@abslineR}{\the\absline@numR+1}
2423       \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@dpagfulltrue\
l@dsamepagefalse}{
2424       \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@dpagfullfalse\
l@dsamepagetrue}{
2425       }{
2426     }
2427   }

```

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

```

2428 \newcommand{\checkverseL}{
2429   \ifinstanzaL
2430   \iflednopbinverse
2431   \ifinserthangingsymbol
2432     \numgdef{\prev@abslineverse}{\the\absline@num-1}
2433     \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{
2434     \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\
prev@abslineverse}\fi}{
2435     \fi
2436   \fi
2437 \fi

```

```

2438 }
2439 \newcommand{\checkverseR}{
2440 \ifinstanzaR
2441   \iflednopbinverse
2442     \ifinserthangingsymbolR
2443       \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2444       \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{}
2445       \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\
prev@abslineverse}\fi}{}
2446     \fi
2447   \fi
2448 \fi
2449 }
2450 %

```

`\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
2451 \newdimen\ledthegoal
2452 \ifshiftedpstarts
2453   \newcommand*{\@goalfraction}{0.95}
2454 \else
2455   \newcommand*{\@goalfraction}{0.9}
2456 \fi
2457 \newcommand*{\check@goal}{%
2458   \ledthegoal=\@goalfraction\pagegoal}
2459 \newcommand{\setgoalfraction}[1]{%
2460   \xdef\@goalfraction{#1}%
2461 }
2462 %
2463 %

```

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

```

\ifwrittenlinesL
2464 \newif\ifwrittenlinesL
2465 \newif\ifwrittenlinesR
2466
2467 %

```

XIX.8 Getting boxes content

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

```

2468 \newcommand*{\get@nextboxL}{%
2469   \ifvbox\namebox{1@dLcolrawbox\the1@dpscl}% box is not empty
2470 %

```

The current box is not empty; do nothing.

```

2471 \else%                                box is empty
2472 %

```

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

```

2473 \ifnum\usernamecount{l@dmxlinesinpar\the\l@dpscl}>\@donetotallinesL
2474 \parledgroup@notes@endL
2475 \else
2476 %

```

Sufficient lines have been output.

```

2477 \ifnum\usernamecount{l@dmxlinesinpar\the\l@dpscl}=\@donetotallinesL
2478 \parledgroup@notes@endL
2479 \fi
2480 \ifwrittenlinesL\else
2481 %

```

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

```

2482 \@writelinesinparL
2483 \writtenlinesLtrue
2484 \fi
2485 \ifnum\l@dnumstartsL>\l@dpscl
2486 %

```

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.

```

2487 \writtenlinesLfalse
2488 \ifbypstart@
2489 \global\line@num=0%
2490 \resetprevline@%
2491 \fi
2492 % Add the content of the optional argument of the previous \protect\cs{pend
2493 % \begin{macrocode}
2494 \csuse{after@pendL@the\l@dpscl}%
2495 \global\csundef{after@pendL@the\l@dpscl}%
2496 %

```

Check the number of lines

```

2497 \l@dcalc@maxoftwo{\the\usernamecount{l@dmxlinesinpar\the\l@dpscl}}%
2498 \the\@donetotallinesL}%
2499 {\usernamecount{l@dmxlinesinpar\the\l@dpscl}}}%
2500 \global\@donetotallinesL \z@
2501 %

```

Go to the next pstart

```

2502 \global\advance\l@dpscl \@ne
2503 \global\pstartnumtrue%
2504 \restore@pstartL@pc%
2505 %

```

Add notes of parallel ledgroup.

```

2506     \parledgroup@notes@endL
2507     \parledgroup@correction@notes@spacing@final{L}
2508     \else
2509 %

2510     \print@last@after@pendLtrue%
2511     \fi
2512     \fi
2513     \fi}
2514 %

2515 \newcommand*{\get@nextboxR}{%
2516     \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2517     \else% box is empty
2518         \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2519             \parledgroup@notes@endR
2520         \else
2521             \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2522                 \parledgroup@notes@endR
2523             \fi
2524             \ifwrittenlinesR\else
2525                 \@writelinesinparR
2526                 \writtenlinesRtrue
2527             \fi
2528             \ifnum\l@dnumstartR>\l@dpscR
2529                 \writtenlinesRfalse
2530             \ifbypstartR
2531                 \global\line@numR=0%
2532                 \resetprevline@%
2533             \fi
2534             \csuse{after@pendR@}\the\l@dpscR}%
2535             \global\csundef{after@pendR@}\the\l@dpscR}%
2536             \l@dcalc@maxoftwo{\the\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}%
2537                 {\the\@donetotallinesR}%
2538                 {\usernamecount{1@dmaxlinesinpar\the\l@dpscR}}}%
2539             \global\@donetotallinesR \z@
2540             \global\advance\l@dpscR \@ne
2541             \global\pstartnumRtrue%
2542             \restore@pstartR@pc%
2543             \parledgroup@notes@endR
2544             \parledgroup@correction@notes@spacing@final{R}
2545         \else
2546             \print@last@after@pendRtrue%
2547         \fi
2548     \fi
2549     \fi}
2550
2551 %

```


XX Page numbering

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

```

2552 \newcommand{\par@patch@thepage}{%
2553   \ifboolexpr{%
2554     bool{sameparallelpagnumber}%
2555     or bool{prevpgnotnumbered}%
2556   }%
2557   {%
2558     \patchcmd{\thepage}%
2559       {page}{par@page}%
2560       {}%
2561       {\led@error@fail@patch@thepage}%
2562   }{}%
2563 }%
2564
2565
2566 \newcommand{\par@patch@pagenumbering}{%
2567   \ifboolexpr{%
2568     bool{sameparallelpagnumber}%
2569     or bool{prevpgnotnumbered}%
2570   }%
2571   {%
2572     \setcounter{par@page}{1}%
2573   }%
2574   {}%
2575   \par@patch@thepage%
2576 }%
2577

```

```

2578 \ifl@dmemoir%
2579   \apptocmd{\@mempnum}%
2580     {\par@patch@pagenumbering}%
2581     {}%%
2582     {\led@error@fail@patch@@mempnum}%
2583
2584 \else%
2585   \apptocmd{\pagenumbering}%
2586     {\par@patch@pagenumbering}%
2587     {}%
2588     {\led@error@fail@patch@pagenumbering}%
2589 \fi%
2590
2591 \AtBeginDocument{\par@patch@thepage}%
2592 %

```

\@outputpage As its name says, \@outputpage is a L^AT_EX'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.

```

2593 \AtBeginDocument{%
2594   \apptocmd{\@outputpage}{%
2595     \ifsameparallelpagenumber%
2596       \ifl@printingpages%
2597         \ifodd\c@page\else%
2598           \stepcounter{par@page}%
2599         \fi%
2600       \else%
2601         \stepcounter{par@page}%
2602       \fi%
2603     \else%
2604       \stepcounter{par@page}%
2605     \fi%
2606   }%
2607   {}%
2608   {\led@error@fail@patch@@outputpage}%
2609 }
2610 %

```

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

```

2611 \newcounter{par@page}%
2612 \setcounter{par@page}{1}%
2613 %

```

XXI Sections' titles' commands

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

aligned.

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

```
\chapter{ 2614 \pretocmd{\chapter}{%
2615 \ifl@dprintingpages%
2616 \vbox{}%
2617 \fi%
2618 }%
2619 {}%
2620 {}%
2621 %
```

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

```
2622 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2623 \eledsectnotoc{R}
2624 %
```

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

```
2625 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2626 \eledsectmark{L}
2627 %
```

`\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`.

```
2628 \newskip\eledsection@correcting@skip
2629 %
```

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

```
2630 \newwrite\eled@sectioningR@out
2631 %
```

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).

```

2632 \def\l@prev@pbR{}
2633 \def\l@prev@nopbR{}
2634 %

```

`\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.

```

2635 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2636 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2637 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2638 \newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\led@nopbnumR{#1}}}
2639 %

```

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

```

2640 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2641 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}
2642 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2643 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}
2644 %

```

XXIII Parallel ledgroup

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

```

2645 \newmarks\parledgroup@
2646 \newmarks\parledgroup@series
2647 \newmarks\parledgroup@type
2648 %

```

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

```

2649 \newcommand{\parledgroup@notes@startL}{%
2650 \ifnum\usernamecount\l@dmxlinesinpar\the\l@dpscl>0%
2651 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
  bhooknoteX}\splitfirstmarks\parledgroup@series}}{%
2652 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
  bhookXnote}\splitfirstmarks\parledgroup@series}}}%
2653 \fi%

```

```

2654 \global\ledgroupnotesL@true%
2655 \insert@noterule@ledgroup{L}%
2656 }
2657 \newcommand{\parledgroup@notes@startR}{%
2658 \ifnum\usernamecount{1@dmxlinesinpar\the\l@dpscR}>0%
2659 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhookeXnoteX\splitfirstmarks\parledgroup@series}}{}%
2660 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookeXnote@\splitfirstmarks\parledgroup@series}}{}%
2661 \fi%
2662 \global\ledgroupnotesR@true%
2663 \insert@noterule@ledgroup{R}%
2664 }
2665 %

```

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

```

2666 \newcommand{\parledgroup@notes@endL}{%
2667 \global\ledgroupnotesL@false%
2668 }
2669 \newcommand{\parledgroup@notes@endR}{%
2670 \global\ledgroupnotesR@false%
2671 }
2672 %

```

`\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`

```

2673 \newcommand{\insert@noterule@ledgroup}[1]{
2674 \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2675 \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
2676 \csuse{ifledgroupnotes#1@}
2677 \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
2678 \csuse{\splitbotmarks\parledgroup@series footnoterule}
2679 \fi
2680 }
2681 {}
2682 \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
2683 \csuse{ifledgroupnotes#1@}
2684 \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
2685 \csuse{footnoterule\splitbotmarks\parledgroup@series}
2686 \fi
2687 }{}
2688 }
2689 {}
2690 }
2691 %

```

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

```
2692 \newcommand{\setparledgroupnotespacing}[1]{\gdef\@parledgroupnotespacing
    {#1}}
2693 \newcommand{\@parledgroupnotespacing}{}
2694 %
```

`\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`.

```
2695 \dimdef{\parledgroup@notespacing@correction}{0pt}
2696 \newcommand{\parledgroup@notespacing@set@correction}{%
2697   {\@getfirstseries\csuse{Xnotefontsize@\@firstseries}%We suppose all the
    series has the same footnote size setup
2698   \@parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
2699   \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing
    }%
2700 }
2701 %
```

`\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.

```
2702 \newcommand{\parledgroup@correction@notespacing@init}{
2703   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
2704   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
2705 }
2706 \parledgroup@correction@notespacing@init
2707 %
```

`\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.

```
2708 \newcommand{\parledgroup@correction@notespacing@final}[1]{
2709   \ifparledgroup
2710     \vspace{\parledgroup@notespacing@correction@accumulated}
2711     \parledgroup@correction@notespacing@init%
2712     \ifstrequal{#1}{L}{
2713       \numdef{\@checking}{\the\l@dpscL-1}
2714     }{
2715       \numdef{\@checking}{\the\l@dpscR-1}
2716     }
2717     \dimdef{\@beforenotes@current@diff}{\csuse{\parledgroup@beforenotes@
    \@checking L}-\csuse{\parledgroup@beforenotes@\@checking R}}%
2718     \ifstrequal{#1}{L}%
```

```

2719     {% Left
2720       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\
@beforenotes@current@diff}}%
2721     }%
2722     {% Right
2723       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\
@beforenotes@current@diff}}{}
2724     }%
2725     \fi
2726   }
2727   %

```

`\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.

```

2728 \newcommand{\parledgroup@correction@notespacing}[1]{%
2729   \csuse{ifledgroupnotes#1@}%
2730   \vspace{-\parledgroup@notespacing@correction}%
2731   \dimdef{\parledgroup@notespacing@correction@accumulated}{\
parledgroup@notespacing@correction@accumulated+\
parledgroup@notespacing@correction}%
2732   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo+\
parledgroup@notespacing@correction}%
2733   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip
}{\advance\numpagelinesL -\@ne%
2734   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo-\baselineskip}%
2735   }% mean greater than equal
2736   \fi%
2737 }
2738 %

```

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

```

2739 \dimdef\parledgroup@beforenotesL{0pt}
2740 \dimdef\parledgroup@beforenotesR{0pt}
2741 %

```

`\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.

```

2742 \newcommand{\parledgroup@beforenotes@save}[1]{
2743   \ifparledgroup
2744     \csdimgdef{@parledgroup@beforenotes@the\csuse{1@dnumstarts#1}#1}{\
csuse{parledgroup@beforenotes#1}}
2745     \csdimgdef{parledgroup@beforenotes#1}{0pt}
2746   \fi
2747 }
2748 %

```

XXIV Compatibility with eledmac

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

```

2749 \ifeledmaccompat%
2750
2751
2752   \unless\ifnocritical@
2753   \let\onlyXside\Xonlyside
2754   \fi
2755 \fi
2756 %

```

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 `\setstanzaindents` 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.

References

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

Index

Symbols

<code>\@adv</code>	1
<code>\@astanza@line</code>	1
<code>\@cs@linesinparL</code>	1
<code>\@cs@linesinparR</code>	1
<code>\@cs@linesonpageL</code>	1
<code>\@cs@linesonpageR</code>	1
<code>\@donereallinesL</code>	1
<code>\@donereallinesR</code>	1
<code>\@donetotallinesL</code>	1
<code>\@donetotallinesR</code>	1
<code>\@eledsectionL</code>	1
<code>\@eledsectionR</code>	1
<code>\@lab</code>	1
<code>\@lopL</code>	1
<code>\@lopR</code>	1

\@nl	1
\@nl@regR	1
\@outputpage	1
\@parledgroupnotespacing	1
\@pend	1
\@pendR	1
\@pstartsfalse	1
\@pstartstrue	1
\@ref	1
\@ref@regR	1
\@set	1
\@stopastanza	1
\@writelinesinparL	1
\@writelinesinparR	1
\@writelinesonpageL	1
\@writelinesonpageR	1
CLASSmemoir	97
COMMAND\@Rlineflag	60, 65
COMMAND\@adv	30, 128
COMMAND\@cs@linesinparL	90
COMMAND\@cs@linesonpageL	90
COMMAND\@eledsectionL	74
COMMAND\@eledsectionR	74
COMMAND\@eledsectmark	99
COMMAND\@eledsectnotoc	99
COMMAND\@footnotetext	67
COMMAND\@goalfraction	9, 94
COMMAND\@l@dtmptcnta	58
COMMAND\@lab	29, 65, 128
COMMAND\@lopL	34, 85, 91
COMMAND\@lopR	34, 91
COMMAND\@mempnum	97
COMMAND\@namedef	69
COMMAND\@namuse	69
COMMAND\@nl	29, 35, 65, 128
COMMAND\@nl@regR	29
COMMAND\@outputpage	97, 98
COMMAND\@page	65
COMMAND\@parledgroupnotespacing	102
COMMAND\@pend	34
COMMAND\@pendR	34
COMMAND\@pstartstrue	73
COMMAND\@ref	32, 33, 35, 128
COMMAND\@ref@regR	32
COMMAND\@set	30, 128
COMMAND\@sw	33
COMMAND\AtBeginPairs	7, 40, 126
COMMAND\AtEveryPend	126–128
COMMAND\AtEveryPstart	2, 13, 47, 75, 126–128
COMMAND\AtEveryPstartCall	2, 13, 47, 127

COMMAND\Clear the right lines for \read@linelist	29
COMMAND\Columns	6, 7, 13, 39, 74, 78, 123–126, 128
COMMAND\Columns@print@after@pend	81
COMMAND\Columns@print@before@pstart	81
COMMAND\Lcolwidth	7, 9, 83
COMMAND\Leftsidehook	123
COMMAND\Leftsidehookend	123
COMMAND\Pages	3, 6, 8, 9, 13, 39, 56, 60, 61, 82, 83, 88, 97, 102, 123, 126–129
COMMAND\Rcolwidth	7, 9, 83
COMMAND\Rightsidehook	123
COMMAND\Rightsidehookend	123
COMMAND\Rlineflag	105
COMMAND\Xmaxnotes	10
COMMAND\Xnoteswidthliketwocolumns	8, 126
COMMAND\Xonlyside	10, 60, 106
COMMAND\&	14
COMMAND\absline@numR	28
COMMAND\add@penalties	58
COMMAND\add@penaltiesL	58
COMMAND\advanceline	30, 36, 128
COMMAND\affixline@num	54
COMMAND\affixline@numR	54, 123, 124
COMMAND\affixpstart@num	56
COMMAND\affixpstart@numR	56
COMMAND\affixside@note	66
COMMAND\aftercolumnseparator	8, 78, 125
COMMAND\araw@textfalse	74
COMMAND\araw@texttrue	74
COMMAND\at@begin@pairs	40
COMMAND\autopar	13
COMMAND\ballast@count	58
COMMAND\baselineskip	78, 103
COMMAND\bbf@set@language	70, 71, 128
COMMAND\beforecolumnseparator	8, 78, 125
COMMAND\begin	14
COMMAND\beginnumbering	11–13, 22, 29, 79, 124, 125, 128
COMMAND\beginnumberingR	35
COMMAND\bf	124
COMMAND\bfseries	124
COMMAND\brokenpenalty	58
COMMAND\chapter	99, 123
COMMAND\check@goal	94
COMMAND\check@pstarts	73
COMMAND\checkpageL	85, 91
COMMAND\checkpb@columns	77
COMMAND\checkpbL	93
COMMAND\checkpbR	93
COMMAND\checkraw@text	74
COMMAND\checkverseL	93
COMMAND\checkverseR	93

COMMAND\clear(double)page	89
COMMAND\clearl@dleftpage	89
COMMAND\clearl@drighpage	89
COMMAND\clearpage	89, 127
COMMAND\cleartoevenpage	89
COMMAND\cleartol@devenpage	89
COMMAND\columnrulewidth	7, 78
COMMAND\columns@position	78
COMMAND\columnseparator	7
COMMAND\columnsposition	7, 125
COMMAND\correct@Xfootins@box	61, 63, 127
COMMAND\correct@footinsX@box	61, 127
COMMAND\critext	127
COMMAND\csname	39
COMMAND\displaywidowpenalty	58
COMMAND\do@actions	52
COMMAND\do@actions@fixedcode	123
COMMAND\do@actions@nextR	52
COMMAND\do@actionsR	52, 123
COMMAND\do@ballast	59
COMMAND\do@ballastR	52
COMMAND\do@insidelineLhook	125
COMMAND\do@insidelineRhook	125
COMMAND\do@line	47
COMMAND\do@line(L/R)	50
COMMAND\do@lineL	48, 58, 123, 124
COMMAND\do@lineLhook	123
COMMAND\do@lineR	50, 123–125
COMMAND\do@lineRhook	123
COMMAND\do@lockoff	129
COMMAND\do@lockoffR	31
COMMAND\do@lockon	128
COMMAND\do@lockonR	31
COMMAND\doinsidelineLhook	126
COMMAND\doinsidelineRhook	126
COMMAND\dolineLhook	126
COMMAND\dolineRhook	126
COMMAND\edindex	127
COMMAND\edlabel	124, 127
COMMAND\edtext	32, 35, 36, 126, 127
COMMAND\eled@sectioningR@out	99
COMMAND\eledchapter	127
COMMAND\eledsection	126, 127, 129
COMMAND\eledsection@correcting@skip	99
COMMAND\eledsectmark	16, 99
COMMAND\eledsectnotoc	16, 99
COMMAND\eledxxx	126
COMMAND\end	14
COMMAND\endgraf	46
COMMAND\endlock	36, 128

COMMAND\endnumbering	11, 13, 22, 128
COMMAND\endsub	36, 128
COMMAND\endumbering	11
COMMAND\expandafter	37
COMMAND\extensionchars	21
COMMAND\firstlinenum	12, 125, 129
COMMAND\firstsublinenum	12, 125, 129
COMMAND\fix@page	30, 128
COMMAND\flag@end	35, 123, 126
COMMAND\flag@start	35, 126
COMMAND\flush@notesR	59
COMMAND\footnoteX	37
COMMAND\footnoteXmk	10
COMMAND\footnoteXnomk	10, 37
COMMAND\frontmatter	17
COMMAND\get@nextboxL	124
COMMAND\get@nextboxR	124
COMMAND\getline@numL	51
COMMAND\getline@numR	51
COMMAND\getlinesfrompagelistL	90
COMMAND\getlinesfrompagelistR	90
COMMAND\getlinesfromparlistL	90
COMMAND\getlinesfromparlistR	90
COMMAND\gl@p	38
COMMAND\goalfraction	105
COMMAND\hangingsymbol	105, 124
COMMAND\hfill	77, 78
COMMAND\hidenumbering	12, 128
COMMAND\ifbypage@	128
COMMAND\ifbypstart@R	128
COMMAND\ifdim	77
COMMAND\ifinserthangingsymbol	67
COMMAND\ifinserthangingsymbolR	67
COMMAND\ifl@dpagefull	91, 92
COMMAND\ifl@dpaging	19, 126
COMMAND\ifl@dpairing	19, 123
COMMAND\ifl@dsamelang	125
COMMAND\ifl@dsamepage	91, 92
COMMAND\ifl@pagefull	85
COMMAND\iflledRcol	19
COMMAND\iflledRcol	124
COMMAND\ifnumberedpar@	42
COMMAND\ifnumberingR	124
COMMAND\ifnumberpstart	39
COMMAND\ifpst@rtedL	21, 22, 43, 123
COMMAND\ifpst@rtedR	21
COMMAND\ifsublines@	31
COMMAND\insert@countR	32
COMMAND\insert@noterule@ledgroup	101
COMMAND\insertlines@list	32

COMMAND\insertlines@listR	32
COMMAND\inserts@list	43
COMMAND\inserts@listR	58
COMMAND\l@d@nums	36, 60
COMMAND\l@d@set	30, 36, 128
COMMAND\l@dLcolrawbox	42
COMMAND\l@dLcolrawbox1	72
COMMAND\l@dLcolrawbox2	72
COMMAND\l@dRcolrawbox	42
COMMAND\l@dbfnote	67, 128
COMMAND\l@dcalc@maxoftwo	91
COMMAND\l@dcalc@minoftwo	91
COMMAND\l@dchecklang	123, 126
COMMAND\l@dcsnote	125
COMMAND\l@dleftbox	47, 85, 127
COMMAND\l@dlinenumR	27, 123
COMMAND\l@dlsnote	125
COMMAND\l@dmake@labels	65
COMMAND\l@dmaxlinesinpar	84
COMMAND\l@dmaxlinesinpar1	72
COMMAND\l@dminpagelines	84, 123
COMMAND\l@dnumptartsL	72, 123
COMMAND\l@dprintingcolumnstrue	127
COMMAND\l@dprintingpagestrue	127
COMMAND\l@dpscL	75, 83, 84, 95
COMMAND\l@dpscR	75, 83, 84
COMMAND\l@drsnote	125
COMMAND\l@dsetupmaxlinecounts	72
COMMAND\l@duselanguage	70, 71, 123
COMMAND\l@dzeromaxlinecounts	72
COMMAND\l@prev@nopbR	99
COMMAND\l@prev@pbR	99
COMMAND\labelstarttrue	124
COMMAND\labelref@list	65
COMMAND\labelref@listR	65
COMMAND\lang	70
COMMAND\last@page@numR	30
COMMAND\led	124
COMMAND\led@nopbR	100
COMMAND\led@nopbnumR	100
COMMAND\led@pbR	100
COMMAND\led@pbnumR	100
COMMAND\ledinnerrote	15
COMMAND\ledleftnote	15
COMMAND\lednopb	15, 77, 93
COMMAND\lednopbR	100
COMMAND\lednopbnumR	100
COMMAND\ledouterote	15
COMMAND\ledpb	77, 93
COMMAND\ledpbR	100

COMMAND\ledpbnumR	100
COMMAND\ledrightnote	15
COMMAND\ledsidenote	15
COMMAND\ledstrutL	123
COMMAND\ledstrutR	123
COMMAND\ledthegoal	94
COMMAND\ledtrutL	123
COMMAND\leftlinenumR	27, 123
COMMAND\let	37, 38
COMMAND\line@list@R	33
COMMAND\line@list@stuff	29, 35
COMMAND\line@margin	26
COMMAND\line@marginR	26, 123
COMMAND\line@numR	28
COMMAND\lineation	12, 127
COMMAND\lineation*	12, 25, 126
COMMAND\lineationR	12, 25, 127
COMMAND\linenum@out	65
COMMAND\linenum@outR	34
COMMAND\linenumincrement	12, 125, 129
COMMAND\linenummargin	12, 25, 123, 128
COMMAND\linenumrepR	27, 123
COMMAND\linesinpar@listL	34, 90
COMMAND\linesonpage@listL	34, 90
COMMAND\lock@off	31
COMMAND\lock@on	31
COMMAND\mainmatter	17
COMMAND\makeatletter	50
COMMAND\maxchunks	6, 14, 72, 73
COMMAND\maxhnotesX	10
COMMAND\memorydump	11, 24
COMMAND\n@num	127
COMMAND\new@lineL	35
COMMAND\new@lineR	35
COMMAND\newhookcommand@series	38
COMMAND\newif	127
COMMAND\newpage	88, 89, 127
COMMAND\newseries	39
COMMAND\newseries@	36
COMMAND\newseries@par	36, 38, 39
COMMAND\noeledxxx	126
COMMAND\nomark@	37
COMMAND\normalbfnoteX	123, 128
COMMAND\notesXwidthliketwocolumns	8, 126
COMMAND\num@lines	58
COMMAND\num@lines(R)	42
COMMAND\numberingR	23
COMMAND\numberlinefalse	5
COMMAND\numberonlyfirstinline	124
COMMAND\numberpstartfalse	12

COMMAND\numberstarttrue	12, 124, 129
COMMAND\one@line	42, 67
COMMAND\one@lineR	42
COMMAND\onlyXside	106
COMMAND\onlysideX	10, 60, 128
COMMAND\otherlanguage	129
COMMAND\page@action	31, 128
COMMAND\pagenumbering	97, 129
COMMAND\pagetotal	85, 127
COMMAND\par@line	58
COMMAND\par@line(R)	42
COMMAND\par@patch@pagenumbering	97
COMMAND\par@patch@thepage	97
COMMAND\parledgroup@	100
COMMAND\parledgroup@beforenotes@save	104
COMMAND\parledgroup@beforenotesL	103
COMMAND\parledgroup@beforenotesR	103
COMMAND\parledgroup@correction@notespacing	103
COMMAND\parledgroup@correction@notespacing@final	102
COMMAND\parledgroup@correction@notespacing@init	102
COMMAND\parledgroup@notes@endL	101
COMMAND\parledgroup@notes@endR	101
COMMAND\parledgroup@notes@startL	100
COMMAND\parledgroup@notes@startR	100
COMMAND\parledgroup@notespacing@correction	102, 103
COMMAND\parledgroup@notespacing@correction@accumulated	103
COMMAND\parledgroup@notespacing@correction@modulo	103
COMMAND\parledgroup@notespacing@set@correction	102
COMMAND\parledgroup@series	100
COMMAND\parledgroup@type	100
COMMAND\parledgroupnotespacing	105
COMMAND\parledgrouptrue	15
COMMAND\patchcmd	128
COMMAND\pausenumbering	23, 24
COMMAND\pend	3, 6, 13–15, 39, 42, 43, 45–47, 72, 81, 125, 126, 128, 129
COMMAND\pendL	125, 126
COMMAND\pendR	126
COMMAND\pendS	13
COMMAND\prev@nopbR	100
COMMAND\prev@pbR	100
COMMAND\prevpgstyle	19
COMMAND\print@Xnotes	60
COMMAND\print@Xnotes@forpages	60, 127
COMMAND\print@columnseparator	77, 126
COMMAND\print@eledsectionL	49
COMMAND\print@line	49
COMMAND\print@lineL	49
COMMAND\print@notesX@forpages	127
COMMAND\printlines	60
COMMAND\printlinesR	60, 123

COMMAND\pstart	3, 6, 12–15, 24, 36, 39, 42, 43, 46, 47, 72, 75, 81, 124, 125, 128, 129
COMMAND\pstartL	47, 125
COMMAND\pstartR	47, 124, 125
COMMAND\pstartinfootnote	127
COMMAND\raw@text	72
COMMAND\read@linelist	29, 107, 128
COMMAND\ref@reg	32
COMMAND\ref@regR	32, 128
COMMAND\relax	106
COMMAND\reledmac	128
COMMAND\renewcommand	105
COMMAND\resumenumbering	23, 24, 125
COMMAND\resumenumberingR	126
COMMAND\rightlinenumR	27, 123
COMMAND\section	123
COMMAND\section@num	21
COMMAND\selectlanguage	13, 70, 71
COMMAND\set@line	36, 128
COMMAND\set@line@action	31, 128
COMMAND\setRlineflag	12, 105
COMMAND\setgoalfraction	9, 105
COMMAND\sethangingsymbol	14
COMMAND\setline	30, 36, 128
COMMAND\setlinenum	31, 36, 128
COMMAND\setnoteposition...	79
COMMAND\setparledgroupnotespacing	105, 129
COMMAND\setposition...	79
COMMAND\setprintlines	123
COMMAND\setstanzaindents	7, 14, 105
COMMAND\setwidth...	79
COMMAND\sidenotemargin	15, 126
COMMAND\sidenotemargin*	15, 126
COMMAND\skipnumbering	12, 127
COMMAND\sloppy	7
COMMAND\stanza	6, 7, 12, 14, 41, 67, 124
COMMAND\stanzanumtrue	15
COMMAND\startlock	36, 128
COMMAND\startsub	36, 128
COMMAND\sub@action	31, 129
COMMAND\sub@off	65
COMMAND\sub@on	65
COMMAND\subline@numR	28
COMMAND\sublinenumincrement	12, 125, 129
COMMAND\sublinenumrepR	27, 123
COMMAND\sza@@	14
COMMAND\textheight	10
COMMAND\textwidth	40
COMMAND\thefootnoteX	125
COMMAND\theledlanguageL	70, 71
COMMAND\theledlanguageR	70, 71

COMMAND\thepage	16, 97
COMMAND\thepstartL	12, 124
COMMAND\thepstartR	12, 124
COMMAND\thestanzaL	15
COMMAND\thestanzaR	15
COMMAND\vbox	44
COMMAND\vl@dbfnote	67
COMMAND\vskip	101
COMMAND\vsplit	58
COMMAND\widthliketwocolumns	8, 19
COMMAND\widthliketwocolumnsfalse	8
COMMAND\widthliketwocolumnstrue	8
COMMAND\xright@appenditem	37
COMMAND\xspace	18
COMMAND\xxxfootstart	79
COMMAND\xxxmatter	97
ENVIRONMENTLeftside	41
ENVIRONMENTRightside	41
ENVIRONMENTastanza	14, 67, 68, 106, 129
ENVIRONMENTpages	40
ENVIRONMENTpairs	40
PACKAGEEDMAC	106
PACKAGEEDSTANZA	106
PACKAGEEledmac	37, 60, 127
PACKAGEEledpar	127
PACKAGETABMAC	106
PACKAGEbabel	13, 70, 71
PACKAGEedmac	106
PACKAGEeledmac	4, 72, 104–106, 125, 126, 128
PACKAGEeledpar	4, 5, 10, 26, 105, 106, 125–127
PACKAGEetoolbox	77, 99
PACKAGEledmac	5
PACKAGEledpar	1, 5
PACKAGEMemoir	106
PACKAGEMusixtex	125
PACKAGEpolyglossia	13, 70, 71
PACKAGEreledmac	1, 3, 5, 6, 8–10, 12–16, 18, 19, 21, 22, 26, 28–32, 34–39, 49, 56, 65, 67, 83, 88, 105, 106, 128, 129
PACKAGEreledpar	1, 3–6, 8, 9, 14–19, 24, 28, 34, 36–39, 65, 105, 106, 128
PACKAGEsetspace	2, 16
PACKAGExkeyval	18

A

\absline@numR	1
\actionlines@listR	1
\actions@listR	1
\add@inserts@nextR	1
\add@insertsR	1
\add@penaltiesL	1
\add@penaltiesR	1

<code>\advanceline</code>	1
<code>\affixline@numR</code>	1
<code>\affixpstart@numL</code>	1
<code>\affixpstart@numR</code>	1
<code>\affixside@noteR</code>	1
<code>\aftercolumnseparator</code>	1, 8
<code>\araw@textfalse</code>	1
<code>\araw@texttrue</code>	1
<code>astanza (environment)</code>	14
<code>\AtBeginPairs</code>	1, 7
<code>\AtEveryPstartCall</code>	1
<code>\autopar</code>	13

B

<code>\bbl@set@language</code>	1
<code>\beforecolumnseparator</code>	1, 8
<code>\beginnumbering</code>	11
<code>\beginnumberingR</code>	1

C

<code>\c@firstlinenumR</code>	1
<code>\c@firstsublinenumR</code>	1
<code>\c@linenumincrementR</code>	1
<code>\c@sublinenumincrementR</code>	1
<code>\ch@ck@l@ckR</code>	1
<code>\ch@cksub@l@ckR</code>	1
<code>\chapter</code>	1
<code>\chapterinpages</code>	1
<code>\check@goal</code>	1
<code>\check@pstarts</code>	1
<code>\checkpageL</code>	1
<code>\checkpageR</code>	1
<code>\checkpb@columns</code>	1
<code>\checkpbL</code>	1
<code>\checkpbR</code>	1
<code>\checkraw@text</code>	1
<code>\checkverseL</code>	1
<code>\checkverseR</code>	1
<code>\clearl@dleftpage</code>	1
<code>\clearl@drightpage</code>	1
<code>\cleartoevenpage</code>	1
<code>\cleartol@devenpage</code>	1
<code>\columnrulewidth</code>	1, 7
<code>\Columns</code>	1, 6
<code>\columns@position</code>	1
<code>\Columns@print@after@pend</code>	1
<code>\Columns@print@before@pstart</code>	1
<code>\columnseparator</code>	1, 7
<code>\columnspan</code>	1, 7
<code>\correct@footinsX@box</code>	1

<code>\correct@Xfootins@box</code>	1
<code>\countLline</code>	1
<code>\countRline</code>	1
<code>\critext</code>	1

D

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

E

<code>\edlabel</code>	1
<code>\edtext</code>	1
<code>\eled@sectioningR@out</code>	1
<code>\eledpar@error</code>	1
<code>\eledsection@correcting@skip</code>	1
<code>\eledsectmark</code>	1, 16
<code>\eledsectnotoc</code>	1, 16
<code>\endlock</code>	1
<code>\endnumbering</code>	1, 11
<code>\endnumberingR</code>	1
<code>\endsub</code>	1
environments:	
<code>astanza</code>	14
<code>Leftside</code>	11
<code>pages</code>	8
<code>pairs</code>	6
<code>Rightside</code>	11

F

<code>\f@x@l@cksR</code>	1
<code>\finish@Pages@notes</code>	1
<code>\first@linenum@out@Rfalse</code>	1

<code>\first@linenum@out@Rtrue</code>	1
<code>\firstlinenum</code>	<u>1</u> , 12
<code>\firstlinenum*</code>	<u>1</u> , 12
<code>\firstsublinenum</code>	<u>1</u> , 12
<code>\firstsublinenum*</code>	<u>1</u> , 12
<code>\fix@page</code>	1
<code>\flag@end</code>	1
<code>\flag@start</code>	1
<code>\flush@notesR</code>	1
<code>\footnoteXmk</code>	10
<code>\footnoteXnomk</code>	10

G

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

H

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

I

<code>\if@pstarts</code>	1
<code>\ifaraw@text</code>	1
<code>\iffirst@linenum@out@R</code>	1
<code>\ifinstanzaL</code>	1
<code>\ifinstanzaR</code>	1
<code>\ifl@dpagfull</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dsamepage</code>	1
<code>\ifl@dusedbabel</code>	1
<code>\ifledRcol</code>	1
<code>\ifprevpgnotnumbered</code>	1
<code>\ifprint@last@after@pendL</code>	1
<code>\ifprint@last@after@pendR</code>	1
<code>\ifpst@rtedL</code>	1
<code>\ifpst@rtedR</code>	1
<code>\ifpstartnumR</code>	1
<code>\ifsameparallelpagenunder</code>	1
<code>\ifshiftedpstarts</code>	1
<code>\ifwidthliketwocolumns</code>	1
<code>\ifwritenlinesL</code>	1
<code>\init@series@par</code>	1
<code>\initnumbering@sectcountR</code>	1
<code>\insert@countR</code>	1

<code>\insert@noterule@ledgroup</code>	1
<code>\inserthangingsymbolL</code>	1
<code>\inserthangingsymbolR</code>	1
<code>\insertlines@listR</code>	1
<code>\inserts@listR</code>	1

L

<code>\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>\lepscL</code>	1
<code>\lepscR</code>	1
<code>\labelref@listR</code>	1
<code>\last@page@numR</code>	1
<code>\Lcolwidth</code>	1, 7, 9
<code>\led@err@BadLeftRightPstarts</code>	1
<code>\led@err@Columns@InsideEnv</code>	1
<code>\led@err@LeftOnRightPage</code>	1
<code>\led@err@Leftside@PreviousNotPrinted</code>	1
<code>\led@err@Pages@InsideEnv</code>	1
<code>\led@err@RightOnLeftPage</code>	1
<code>\led@err@Rightside@PreviousNotPrinted</code>	1
<code>\led@err@TooManyPstarts</code>	1
<code>\led@error@fail@patch@@mempnum</code>	1
<code>\led@error@fail@patch@@outputpage</code>	1
<code>\led@error@fail@patch@@pagenumbering</code>	1
<code>\led@error@fail@patch@thepage</code>	1
<code>\led@nopbnumR</code>	1

<code>\led@nopbR</code>	1
<code>\led@pbnumR</code>	1
<code>\led@pbR</code>	1
<code>\lednopbnum</code>	1
<code>\lednopbnumR</code>	1
<code>\ledpbnumR</code>	1
<code>\ledpbR</code>	1
<code>\ledstrutL</code>	1
<code>\ledstrutR</code>	1
<code>\ledthegoal</code>	1
<code>\leftlinenumR</code>	1
<code>\leftpstartnumL</code>	1
<code>\leftpstartnumR</code>	1
<code>Leftside (environment)</code>	11
<code>\Leftsidehook</code>	1
<code>\Leftsidehookend</code>	1
<code>\line@list@stuffR</code>	1
<code>\line@listR</code>	1
<code>\line@marginR</code>	1
<code>\line@numR</code>	1
<code>\lineation*</code>	1, 12
<code>\lineationR</code>	1, 12
<code>\linenum@outR</code>	1
<code>\linenumincrement</code>	1, 12
<code>\linenumincrement*</code>	1, 12
<code>\linenummargin</code>	1
<code>\linenumrepR</code>	1
<code>\linesinpar@listL</code>	1
<code>\linesinpar@listR</code>	1
<code>\list@clearing@regR</code>	1
<code>\list@pstartL@pc</code>	1
<code>\list@pstartR@pc</code>	1
<code>\lock@off</code>	1

M

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

N

<code>\n@num</code>	1
<code>\namebox</code>	1
<code>\new@lineL</code>	1
<code>\new@lineR</code>	1
<code>\newnamebox</code>	1
<code>\newnamecount</code>	1
<code>\newseries@par</code>	1
<code>\normalbfnoteX</code>	1

<code>\notesXwidthliketwocolumns</code>	8
<code>\num@linesR</code>	1
<code>\numberpstartfalse</code>	12
<code>\numberpstarttrue</code>	12
<code>\numpagelinesL</code>	1
<code>\numpagelinesR</code>	1

O

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

P

<code>\page@action</code>	1
<code>\page@numR</code>	1
<code>\Pages</code>	1, 8
<code>pages</code> (environment)	8
<code>pairs</code> (environment)	6
<code>\par@lineR</code>	1
<code>\par@patch@pagenumbering</code>	1
<code>\par@patch@thepage</code>	1
<code>\parledgroup@</code>	1
<code>\parledgroup@beforenotes@save</code>	1
<code>\parledgroup@beforenotesL</code>	1
<code>\parledgroup@beforenotesR</code>	1
<code>\parledgroup@correction@notespacing</code>	1
<code>\parledgroup@correction@notespacing@final</code>	1
<code>\parledgroup@correction@notespacing@init</code>	1
<code>\parledgroup@notes@startL</code>	1
<code>\parledgroup@notes@startR</code>	1
<code>\parledgroup@notespacing@correction</code>	1
<code>\parledgroup@notespacing@set@correction</code>	1
<code>\parledgroupseries@</code>	1
<code>\parledgrouptype@</code>	1
<code>\pausenumberingR</code>	1
<code>\pend</code>	13
<code>\pendL</code>	1
<code>\pendR</code>	1
<code>\prev@nopbR</code>	1
<code>\prev@pbR</code>	1
<code>\prevpgstyle</code>	1
<code>\print@columnseparator</code>	1
<code>\print@eledsectionL</code>	1
<code>\print@eledsectionR</code>	1
<code>\print@lineL</code>	1
<code>\print@lineR</code>	1
<code>\print@notesX@forpages</code>	1
<code>\print@Xnotes@forpages</code>	1
<code>\printlinesR</code>	1
<code>\pstart</code>	13
<code>\pstartL</code>	1

\pstartR 1

R

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

S

\section@numR 1
 \selectlanguage 1
 \set@line 1
 \set@line@action 1
 \setgoalfraction 1, 9
 \sethangingsymbol 14
 \setline 1
 \setlinenum 1
 \setnamebox 1
 \setnote@position@liketwocolumns@C 1
 \setnote@position@liketwocolumns@L 1
 \setnote@position@liketwocolumns@R 1
 \setposition@liketwocolumns@C 1
 \setposition@liketwocolumns@L 1
 \setposition@liketwocolumns@R 1
 \setRlineflag 12
 \setwidth@liketwocolumns@C 1
 \setwidth@liketwocolumns@L 1
 \setwidth@liketwocolumns@R 1
 \sidenote@marginR 1
 \sidenotemargin* 1
 \skip@lockoff 1
 \skipnumbering 1, 12
 \startlock 1
 \startsub 1
 \sub@action 1
 \subline@numR 1
 \sublinenumincrement 1, 12
 \sublinenumincrement* 1, 12
 \sublinenumrepR 1

T

\theledlanguageL 1

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

U

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

W

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

X

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

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	70
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	75
\Pages: Added \l@duselanguage to \Pages	85
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	50
Added hooks into Leftside environment	41
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match neweledmac	54
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	52
\do@lineL: Added \do@lineLhook to \do@lineL	48
Simplified \do@lineL by using macros for some common code	48
\do@lineR: Changed \do@lineR similarly to \do@lineL	50
\flag@end: Removed extraneous spaces from \flag@end	35
\ifledRcol: Moved \ifl@dpairing toeledmac	19
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac	21
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	27
\l@dnumpstartsR: Moved \l@dnumpstartsL toeledmac	72
\ledstrutR: Added \ledstrutL and \ledstrutR	89
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX	67
\Pages: Added \ledstrutL to \Pages	85
Added \ledstrutR to \Pages	86
\printlinesR: Simplified \printlinesR by using \setprintlines	60
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	41
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	27
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Do not just set \line@marginR in \linenummargin	26
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	87
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>	12
Possibility to number the pstart with the commands <code>\numberpstarttrue</code>	1
<code>\ifledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code>	19
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>).	48
Lineation can be by pstart (like in <code>eledmac 0.15</code>).	24
New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . .	67
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code>).	54
<code>\Columns</code> : Line numbering by pstart.	76
<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>).	94
Pstart number can be printed in side	95
<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	67
v0.12.0.	
General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	67
v1.0.0.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code>	1
Debug in lineation by pstart	24
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>).	43
v1.1.1.	
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1.	43
v1.1.2.	
<code>\affixside@noteR</code> : Remove spurious space between line number and line content . .	66
v1.2.0.	
General: Support for <code>\led<section></code> commands in parallel texts.	1
v1.2.1.	
<code>\initnumbering@sectcountR</code> : For the right section, the counter is defined only once.	23

v1.3.0.	
\edtext: Manage RTL language.	36
v1.3.1.	
\l@dbfnote: Compatibility of standard footnotes with eledmac when theses footnotes contain any commands.	67
v1.3.2.	
General: Debug with some classes.	1
v1.3.3.	
General: Debugging the left notes of the right column.	66
\l@dbfnote: Spurious space with footnote in right column.	67
v1.3.4.	
General: Allow use of commands in sidenotes, as introduced by eledmac 1.0.	66
v1.3.5.	
\normalbfnoteX: Allows one to redefine \thefootnoteX with alph when some packages are loaded.	67
v1.4.0.	
General: Added \do@insidelineLhook and \do@insidelineRhook	50
v1.4.1.	
General: Enable the use of stanzaindentsrepetition within astanza environment.	67
\normalbfnoteX: Fix bug with normal familiar footnotes when mixing RTL and LTR text.	67
v1.4.3.	
General: Corrects a false hanging verse when a verse is exactly the length of a line.	1
\inserthangingsymbolR: Hanging verse is no longer automatically flush right.	67
\pendL: Spurious spaces in \pendL.	45
\pendR: Spurious spaces in \pstartR.	46
\pstartR: Spurious spaces in \pstartL and \pstartR.	43
v1.5.0.	
General: Add, as in eledmac, features to manage page breaks.	1
\sublinenumincrement*: Add starred version of \firstlinenum, \linenumincrement, \firstsublinenum, \sublinenumincrement to change both Left and Rightside.	26
v1.6.0.	
General: Add tool and documentation for parallel ledgroups	15
v1.7.0.	
General: Add, as in eledmac, features to make crossrefs with pstart numbers.	1
v1.8.0.	
General: \beginnumbering is defined only on eledmac, not on eledpar.	22
\l@dlsnote, \l@drsnote and \l@dcnote defined only one time, in eledmac.	66
Add \beforecolumnseparator and \aftercolumnseparator.	8
Add \columnspostion.	7
Add, as in eledmac, new system of sectioning commands.	1
Add, as in eledmac, option to insert something after \pends / verses.	1
Add, as in eledmac, option to insert something between \pstarts / verse.	1
Change \do@lineR and \do@lineR to allow new sectioning commands.	48
Compatibility with musixtex.	1
Debug eledmac sectioning command after using \resumenummering.	1
New sectioning commands, as in eledmac.	16
Suppress \ifl@dsamelang which did not work and was not logical, because both columns could have the same language but not the main language of the document.	70
\Columns: Modify \Columns to enable to add section's title.	74

Suppress \l@dchecklang from \Columns.	75
\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.	70
\Pages: Modify \Pages to enable to add section's title.	82
\pendL: As in eledmac, \pendL can have an optional argument.	45
\pendR: As in eledmac, \pendR can have an optional argument.	46
\print@columnseparator: Move some code of \Columns to \print@columnseparator.	77
\pstartR: As in eledmac, \pendL and \pendR can have an optional argument.	43
\sidenotemargin*: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar.	65
Add \sidenotemargin*	65
\theledlanguageR: Correct left/right language setting with polyglossia.	71
v1.8.1.	
\do@lineL: Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?).	48
\do@lineR: Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?).	50
v1.8.2.	
General: Debug \eledxxx with some paper sizes	1
Debug left and side note (bugs added by 1.8.0)	1
\eledpar@error: Errors specific to eledpar send to eledpar handbook	20
\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar	35
\lineation*: Add \lineation*	25
v1.8.3.	
General: Add \noeledxxx, as in eledmac	1
\doinsidelineRhook: Added \dolineLhook, \dolineRhook, \doinsidelineLhook and \doinsidelineRhook	50
\Pages: Debug blank pages when using optional argument in the last \pend.	82
\resumenumberingR: Debug \resumenumberingR	23
v1.9.0.	
General: Add \AtBeginPairs macro.	7
Compatibility with \Xnoteswidthliketwocolumns and \notesXwidthliketwocolumns	1
\ifwidthliketwocolumns: Added widthliketwocolumns option	19
\theledlanguageR: Debug left/right language switching with polyglossia. Do not write in .aux file when setting left/right lines.	71
v1.9.1.	
\ifledRcol: Moved \ifl@dpaging to eledmac	19
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).	82
Debug wrong parallel pages synchronization when an \edtext falls across two pages.	82
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.	35

v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles.	1
\edtext: \critext and \edtext are now defined only in eledmac.	36
v1.12.0.	
General: Compatibility with Lua \TeX RTL languages.	1
\Columns: Add \l@dprintingcolumnstrue	74
\edlabel: \edlabel and \edindex works now with hyperref when using eledpar. . .	65
\edlabel is now defined only one time for both eledmac and eledpar	65
\Pages: Add \l@dprintingpagestrue	82
\print@eledsectionL: Compatibility with Lua \TeX RTL languages.	49
\print@eledsectionR: Compatibility with Lua \TeX RTL languages.	51
\print@lineL: Compatibility with Lua \TeX RTL languages.	49
v1.12.1.	
\print@eledsectionL: Fixes bug with Lua \TeX RTL \eledsection.	49
v1.13.0.	
General: Enable the use of optional argument of & in astanza environment.	67
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.	89
\ifledRcol: Remove false boolean settings which are not needed.	19
\Pages: Prevent false overfull hboxes when using \Pages outside of pages environment. When using shiftedpstarts option, a \l@dleftbox with a null height will advance the \pagetotal in any case.	83 82
v1.13.1.	
\correct@footinsX@box: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages.	60
Correct \correct@footinsX@box and \correct@Xfootins@box	60
\Pages: Prevent false empty page after \Pages (bug added in 1.13.0)	82
v1.14.0.	
General: Fix bug with line number position when using \eledsection and similar com- mands for RTL texts with Lua \TeX	1
The \newifs are not followed by boolean values set to false, because it is the \TeX default setting.	1
v1.15.0.	
General: Add \AtEveryPstartCall.	1
Add sameparallepagenumber option.	9
Fix vertical spurious space before right \eledchapter (bug added in v1.13.0).	1
Prevent vertical space when using \AtEveryPstart or \AtEveryPend with a com- mand which prints nothing	1
\do@actions@nextR: Add action 1008 and 1009	52
\inserthangingsymbolR: Prevent more efficiently the column separator from shifting when a verse is hanging	67
\lineationR: As \lineation, \lineationR automatically set the \pstartinfootnote.	25
\n@num: \n@num defined only one time for both Eledmac and Eledpar.	32
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar	36

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.	37
v1.16.1.	
General: Write information about line-list file version in the correct file.	1
v1.16.2.	
General: Fix bug when adding empty lines before a \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	12
v2.0.0.	
\@adv: \@adv defined only in reledmac.	30
\@lab: \@lab defined only in eledmac.	65
\@ref@regR: \@ref defined only in reledmac, code specific to right side moved in \ref@regR.	32
\@set: \@set defined only in reledmac.	30
General: \@nl is now defined only in reledmac.	29
\ifbypage@ and \ifbypstart@R defined in eledmac.	24
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.	36
\bbl@set@language: Patch \bbl@set@language instead of redefining it	70
\do@lockonR: \do@lockon defined only in reledmac.	31
\endlock: \startlock and \endlock defined only in reledmac.	36
\endsub: \startsub and \endsub defined only in reledmac.	36
\fix@page: \fix@page is defined only once in reledmac	30
chapterinpages: Deleting the old system of managing parallel chapter, keep only the new one with \patchcmd.	40
\l@d@set: \l@d@set defined only in reledmac.	31
\l@dbfnote: \l@dbfnote defined only in reledmac.	67
\line@marginR: \linenummargin now defined only once time in reledmac.	26
\normalbfnoteX: \normalbfnoteX defined only in reledmac.	67
\page@action: \page@action defined only in reledmac.	31
\read@linelist: \read@linelist is defined only once time in \reledmac.	29
\set@line: \set@line defined only in reledmac.	36
\set@line@action: \set@line@action defined only in reledmac.	31
\setline: \setline defined only in reledmac.	36
\setlinenum: \setlinenum defined only in reledmac.	36

<code>\skip@lockoff: \do@lockoff</code> defined only in <code>reledmac</code>	31
<code>\sub@action: \sub@action</code> defined only in <code>reledmac</code>	31
<code>\sublinenumincrement*: \firstlinenum, \linenumincrement, \firstsublinenum,</code> <code>\sublinenumincrement</code> are now defined only in <code>reledmac</code>	26
<code>\theledlanguageR: Patch \otherlanguage</code> instead of redefining it.	71
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