

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	6
2 Options	6
2.1 Synchronization’s options	6
2.2 Other options	6
3 General	7

*This file (`reledpar.dtx`) has version number v2.6.1, last revised 2015/11/20.

[†]maieul at maieul dot net

[‡]herries dot press at earthlink dot net

4 Parallel columns	8
4.1 Basic use	8
4.2 Setting	8
4.2.1 Column's width	8
4.2.2 Column's separator	9
4.2.3 Column's positions	9
4.2.4 Mixing two columns and one column texts	9
5 Facing pages	9
5.1 Basic usage	9
5.2 Setting	10
5.2.1 Text width	10
5.2.2 Way of synchronizing	10
5.2.3 Page number	12
5.2.4 Page breaking	12
5.2.5 Right page before \Pages	12
5.2.6 Notes about \mainmatter	12
5.3 Critical and familiar footnotes	13
5.3.1 Notes height setting	13
5.3.2 About the numbering of familiar footnotes	13
5.3.3 Using perpage package	13
5.3.4 Notes for one side only	14
5.3.5 Familiar notes called in the right side, but to be printed in the left side	14
5.4 Using line flag	14
6 Left and right texts	15
6.1 Environments	15
6.2 Numbering text lines and paragraphs	15
6.3 Line numbering scheme	16
6.4 Lineation system	16
6.5 Chunks	17
6.6 \AtEveryPstart and \AtEveryPstartCall	17
6.7 Language setting	17
7 Verse	18
8 Side notes	19
9 Parallel ledgroups	19
9.1 General	19
9.2 Parallel ledgroups and setspace package	20
10 Sectioning commands	20
11 Notes about page number	20
I Implementation overview	21

II Preliminaries	21
II.1 Package's meta-data	21
II.2 Package's requirement	21
II.3 Package's options	21
II.4 Package's options	22
II.4.1 Synchronization's options	22
II.4.2 Other options	23
II.5 Determining side and category of parallel processing	23
II.6 Text's width	24
II.7 Messages	24
III Sectioning commands	26
IV Line counting	30
IV.1 Setting lineation reset	30
IV.2 Setting line number margin	31
IV.3 Setting lineation start and step	31
IV.4 Setting line flag	33
IV.5 Setting line number style	33
IV.6 Print marginal line number	34
IV.7 Line-number counters and lists	34
IV.7.1 Correspond to those in <code>reledmac</code> for regular or left text	34
IV.7.2 Specific to <code>reledpar</code>	35
IV.8 Reading the line-list file	35
IV.9 Commands within the line-list file	36
IV.10 Writing to the line-list file	42
V Marking text for notes	43
V.1 Specific hooks and commands for notes	43
V.1.1 Notes to be printed on one side only	44
V.2 Tools specific to familiar footnotes	44
V.2.1 Managing correct number	44
V.2.2 Familiar footnotes without marks	45
V.2.3 Get correct footnote number	46
V.3 Create hooks	46
V.4 Init standards series (A,B,C,D,E,Z)	47
V.5 Tools specific to \LaTeX 's classical footnotes	47
VI Pstart numbers dumping and restoration	47
VII Parallel environments	48
VIII Paragraph decomposition and reassembly	51
VIII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code>	51
VIII.2 Processing one line	56
VIII.3 Line and page number computation	60
VIII.4 Line number printing	62

VIII.5 Pstart number printing in side	65
VIII.6 Add insertions to the vertical list	66
VIII.7 Penalties	67
VIII.8 Printing leftover notes	68
IX Footnotes	68
IX.1 Footnotes output specific to \Pages	68
X Cross referencing	73
XI Side notes	73
XII Familiar footnotes	75
XIII Verse	75
XIV Naming macros	77
XV Fixing babel and polyglossia	78
XVI Counts and boxes for parallel texts	80
XVII Checking text to be processed	81
XVIII Parallel columns	83
XIX Parallel pages	91
XIX.1 Specific counters	91
XIX.2 Main macro	91
XIX.3 Ensure all notes be printed at the end of parallel pages	97
XIX.4 Struts	98
XIX.5 Page clearing	98
XIX.6 Lines managing	99
XIX.7 Page break managing	101
XIX.8 Getting boxes content	104
XX Page numbering	108
XX.1 Global options	108
XX.2 mainmatter option of \Pages	110
XXI Sections' titles' commands	110
XXII Page break/no page break, depending on the specific line	111
XXIII Parallel ledgroup	112
XXIV Compatibility with eledmac	116

XXV The End	116
Appendix A Some things to do when changing version	117
Appendix A.1 Migration to eledpar 1.4.3	117
Appendix A.2 Migration from eledpar to reledpar	117
Appendix A.2.1 Deprecated options	117
Appendix A.2.2 \renewcommandreplaced with command	117
Appendix A.2.3 Commands the names of which have changed	118
Appendix A.3 Migration to reledpar 2.2.0	118
Appendix A.4 Migration to reledpar 2.3.0	118
Appendix A.5 Migration to reledpar 2.4.0	118
Appendix A.6 Migration to reledpar 2.5.0	118
Appendix A.7 Migration to reledpar 2.6.0	118
Appendix A.8 Migration to reledpar 2.6.1	118
References	119
Index	119
Change History	137

1 Introduction

1.1 Aim of this package

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

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

This manual contains a general description of how to use `reledpar` starting in section 3; the complete source code for the package, with extensive documentation (in sections I through XXV); and an Index to the source code. As `reledpar` is an adjunct to `reledmac` we assume that you have read the `reledmac` manual. Also `reledpar` requires `reledmac` to be used, in the version distributed with version.

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

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

1.2 Historical overview

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

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

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

2 Options

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

2.1 Synchronization's options

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

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

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

nomaxlines allows facing pages to have different numbers of lines.

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

2.2 Other options

parledgroup allows the use of `ledgroup` environment with `reledpar`.¹

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

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

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

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

²This option can either be used on `reledmac` or `reledpar`.

3 General

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

The `reledpar` package lets you typeset two *numbered* texts in parallel³. This can be done either as setting the 'Leftside' and 'Rightside' texts in two columns or on facing pages. In the paired pages case footnotes are placed at the bottom of the page on which they are called out — that is, footnotes belonging to the left are set at the foot of a left (even numbered) page, and those for right texts are at the bottom of the relevant right (odd numbered) page. However, in the columnar case, all footnotes are set at the bottom left of the page on which they are called out — they are not set below the relevant column. The line numbering schemes need not be the same for the two texts.

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

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

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

```
\maxchunks{5120}
```

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

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

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

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

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

4 Parallel columns

4.1 Basic use

`pairs` Numbered text that is to be set in columns must be within a `pairs` environment. Within the environment the text for the lefthand and righthand columns is placed within the `Leftside` and `Rightside` environments, respectively; these are described in more detail below in section 6.

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

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

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

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

There is no required pagebreak before or after the columns.

4.2 Setting

4.2.1 Column's width

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

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

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

4.2.2 Column’s separator

`\columnrulewidth` `\columnseparator` The macro `\columnseparator` is called between each left/right pair of lines. By default it inserts a vertical rule of width `\columnrulewidth`. As this is initially defined to be 0pt the rule is invisible. For a visible rule between the columns you could try:
`\setlength{\columnrulewidth}{0.4pt}`
 You can also modify `\columnseparator` if you want more control.

4.2.3 Column’s positions

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

4.2.4 Mixing two columns and one column texts

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

`\Xnoteswidthliketwocolumns` `\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.

5 Facing pages

5.1 Basic usage

`pages` Numbered text that is to be set on facing pages must be within a `pages` environment.

Within the environment the text for the lefthand and righthand pages is placed within the `Leftside` and `Rightside` environments, respectively.

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

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

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

5.2 Setting

5.2.1 Text width

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

5.2.2 Way of synchronizing⁴

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

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

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

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

The aim of the default setting is twofold:

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

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

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

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

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

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

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

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

5.2.3 Page number

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

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

Please also read advising in 11 p. 20.

5.2.4 Page breaking

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

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

5.2.5 Right page before `\Pages`

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

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

`prevpgstyle=<style>` in order to set the style of this page. A common value of `<style>` is empty. Use `prevpgstyle=empty` will suppress header and footer in this page. Please also read advising in 11 p. 20.

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

5.2.6 Notes about `\mainmatter`

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

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

5.3 Critical and familiar footnotes

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

5.3.1 Notes height setting

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

However, the `reledmac` default setting for the maximum allotted size to notes is greater than `\textheight`. That makes impossible for long notes to flow across pages.⁵ We have not changed this default setting, because we do not want to break compatibility with older version of `reledmac` and we want to be as close as possible to default \LaTeX ’s feature.

So, you MUST change the default setting via `\Xmaxhnotes` (for critical notes) and `\maxhnotesX` (for familiar notes). Both commands are explained in `reledmac` handbook (6.11.5 p. 39). As an advisable setting:

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

5.3.2 About the numbering of familiar footnotes

If you use the same series of familiar footnotes on both sides, the numbers won’t be correct in the first run. There will be a continuous numbering for left notes, and a continuous numbering for right notes. However, after the second run, the numbering will be continuous, alternating between the left and right side. For example if you have two left pages and two right pages, with one note by page, you will obtain the following numbering at the first run: 1 (left page), 3 (right page), 2 (left page), 4 (right page). But at the next run, you will obtain: 1 (left page), 2 (right page), 3 (left page), 4 (right page).

If you use parallel columns, during the second of run of typesetting the footnote numbering will not run down the columns. Instead, it will read both column lines completely across the page, and number footnotes from left to right.

5.3.3 Using `perpage` package

It follows from what has been said in the preceding paragraph that if you use the `\MakePerPage` command of the `perpage` package for footnotes called in parallel typesetting, you must append to the counter the suffix `@typeset`.

So do not set:

```
\MakePerPage{footnote}
\MakePerPage{footnoteA}
\MakePerPage{footnoteB}
```

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

But set:

```
\MakePerPage{footnote@typeset}
\MakePerPage{footnoteA@typeset}
\MakePerPage{footnoteB@typeset}
```

5.3.4 Notes for one side only

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

5.3.5 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.

`\footnoteXmk`

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\footnoteAmk. And so one ...
\pend
\endnumbering
\end{Rightside}
```

5.4 Using line flag

`\Xlineflag` Use `\Xlineflag[⟨s⟩]` to add right line flag (6.4 p. 16) to right critical footnotes and
`\Xendlineflag` `\Xendlineflag[⟨s⟩]` to add it to right critical endnotes.

6 Left and right texts

6.1 Environments

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

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

6.2 Numbering text lines and paragraphs

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

```
\beginnumbering
<text>
\endnumbering
```

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

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

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

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

```

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

```

```

\numberpstarttrue
\numberpstartfalse
  \thepstartL
  \thepstartR
\skipnumbering
\hidenumbering

```

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

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

6.3 Line numbering scheme

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

6.4 Lineation system

```

\firstlinenum
\linenumincrement
\firstsublinenum
\sublinenumincrement

```

Following `\firstlinenum{<num>}` the first line number will be `<num>`, and following `\linenumincrement{<num>}` only every `<num>`th line will have a printed number.

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

```

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

```

The starred versions change both left and right numbering schemes.

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

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

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

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

`\linenummarginR{<margin>}` sets the line margin for right side. `\linenummargin*{<margin>}` sets for both side. `<margin>` can be, as for `reledmac`’s `\linenummargin` one of these values: left, right, inner, outer. A “R” is appended to the line numbers of the right texts. This may be useful for parallel columns but for parallel pages it might be

more appropriate to redefine it using `\setRlineflag{flag}`. Use `\setRlineflag{}` to empty it.

6.5 Chunks

`\pstart`
`\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. 16).

6.6 \AtEveryPstart and \AtEveryPstartCall

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

Consequently:

- The argument of `\AtEveryPstart` (see 4.2.4 p. 17) is called before every chunk is printed, except if you used an optional argument for the `\pstart`.
- The argument of `\AtEveryPstartCall` is called before every `\pstart`.

6.7 Language setting

If you are using the `babel` package or the `polyglossia` package, with different languages (via, say, `\selectlanguage`) for the left and right texts it is particularly important to select the appropriate language within the `Leftside` and `Rightside` environments. The initial language selected for the right text is the `babel` package's default.

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

7 Verse

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

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

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

The difference between `astanza` and `\stanza` is, that the 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 `\szi@00`’ 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` `\thestanzaR` When using `\stanzanumtrue` (8.9 p. 43) 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.

8 Side notes

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

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

9 Parallel ledgroups

9.1 General

You can also make parallel ledgroups (see the documentation of `reledmac` about ledgroups, 9 p. 44). To do it you have:

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

See the following example:

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

```
\end{pages}
\Pages
```

9.2 Parallel ledgroups and setspace package

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

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

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

10 Sectioning commands

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

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

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

11 Notes about page number

If you use `sameparallepagenumber` option (5.2.3 p. 12 or `prevpgnotnumbered` option (5.2.5 p. 12), please read the following paragraph if you want to manipulate page numbers manually.

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

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

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

I Implementation overview

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

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

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

II Preliminaries

II.1 Package’s meta-data

Announce the name and version of the package, which is targeted for \LaTeX 2e. 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}[2015/11/20 v2.6.1 reledmac extension for
  parallel texts]%
4
5 %
```

II.2 Package’s requirement

Few commands use `\xspace` command.

```

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

II.3 Package’s options

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

```

8 \RequirePackage{xkeyval}
9 %
```

II.4 Package's options

II.4.1 Synchronization's options

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

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

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

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

With the option 'advancedshiftedpstarts' a long pstart on the left side (or on the right side) does not make a blank on the corresponding pstart, but the blank is put on the bottom of the page. Consequently, the pstarts on the parallel pages are shifted, but the shift stops at every end of pages. Differing to shiftedpstarts, the pstart shift are not counted to determine when cutting the page. That could help to avoid page with blank lines at the bottom.

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

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

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

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

```

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

```

II.4.2 Other options

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

```

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

```

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

```

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

```

`\ifsameparallelpagenunder` Options related to page numbering

`\ifprevpgnotnumbered`

```

43 \newif\ifsameparallelpagenunder
44 \newif\ifprevpgnotnumbered
45 \DeclareOptionX{sameparallelpagenunder}{\sameparallelpagenundertrue}
46 \DeclareOptionX{prevpgnotnumbered}{\prevpgnotnumberedtrue}
47 %

```

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

```

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

```

```

50 \ProcessOptionsX%
51 %

```

II.5 Determining side and category of parallel processing

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

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

II.6 Text's width

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

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

II.7 Messages

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

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

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

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

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

```
\led@err@LeftOnRightPage69 \providebool{syntax@}
\led@err@RightOnLeftPage70 \newcommand*\led@err@LeftOnRightPage{%
71 \notbool{syntax@}%
72 {\reledpar@error{The left page has ended on a right page}{\@ehc}}%
73 {}%
74 }
75 \newcommand*\led@err@RightOnLeftPage{%
76 \notbool{syntax@}%
77 {\reledpar@error{The right page has ended on a left page}{\@ehc}}
78 {}%
79 }%
80 %
```



```

ftside@PreviousNotPrinted81 \newcommand*{\led@err@Leftside@PreviousNotPrinted}{%
htside@PreviousNotPrinted82   \reledpar@error{You call a new Leftside environment while the previous
                                one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
83 \newcommand*{\led@err@Rightside@PreviousNotPrinted}{%
84   \reledpar@error{You call a new Rightside environment while the previous
                                one has not been typeset by \string\Pages\space or \string\Columns}{\@ehc}}
85 %

```

```

\led@err@Pages@InsideEnv86 \newcommand*{\led@err@Pages@InsideEnv}{%
led@err@Columns@InsideEnv87   \reledpar@error{\string\Pages\space must be called *outside* of the `
                                pages` environment}{\@ehc}}
88 \newcommand*{\led@err@Columns@InsideEnv}{%
89   \reledpar@error{\string\Columns\space must be called *outside* of the `
                                pairs` environment}{\@ehc}}
90 %

```

```

\led@err@Pages@WithoutEnv91 \newcommand*{\led@err@Pages@WithoutEnv}{%
led@err@Columns@WithoutEnv92   \reledpar@error{\string\Pages\space called without previous `pages`
                                environment}{\@ehc}}
93 \newcommand*{\led@err@Columns@WithoutEnv}{%
94   \reledpar@error{\string\Columns\space called without previous `pairs`
                                environment}{\@ehc}}
95 %

```

```

@error@fail@patch@thepage96 \newcommand{\led@error@fail@patch@thepage}{%
97   \reledpar@error{Fail to patch \string\@thepage\space command.}{\@ehc}%
98 }%
99 %

```

```

@fail@patch@pagenumbering100 \newcommand{\led@error@fail@patch@pagenumbering}{%
101   \reledpar@error{Fail to patch \string\pagenumbering\space command.}{\@ehc
102 }%
103 }%
104 %

```

```

error@fail@patch@@mempnum104 \newcommand{\led@error@fail@patch@@mempnum}{%
105   \reledpar@error{Fail to patch \string@mempnum\space command.}{\@ehc}%
106 }%
107 %

```

```

or@fail@patch@@outputpage108 \newcommand{\led@error@fail@patch@@outputpage}{%
109   \reledpar@error{Fail to patch \string@outputpage\space command.}{\@ehc}%
110 }%
111 %

```

```

\led@warn@ChangeSyncOption 12 \newcommand*{\led@warn@ChangeSyncOption}[1]{%
113 \reledpar@warning{You have changed synchronization's options since last
run. We have not read line-list file #1. Please run LaTeX again.}%
114 }%
115 %

```

```

\led@warn@setting@in@rightside 16 \newcommand{\led@warn@setting@in@rightside}[1]{%
117 \reledpar@warning{You use #1 inside rightside environment.\MessageBreak%
118 Such behavior is deprecated.\MessageBreak%
119 Use instead #1R or #1* in your preamble}.%
120 }
121 %

```

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.

```

122 \newcount\section@numR
123 \section@numR=\z@
124 %

```

`\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`.

```

125 \pst@rtedLfalse
126 \newif\ifpst@rtedR
127
128 %

```

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

```

129 \newcommand*{\beginnumberingR}{%
130 \ifnumberingR
131 \led@err@NumberingStarted
132 \endnumberingR
133 \fi
134 \global\l@dnumpstartsR \z@
135 \global\pst@rtedRfalse
136 \global\numberingRtrue
137 \global\advance\section@numR \@ne
138 \global\absline@numR \z@

```

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

```

161 \def\endnumberingR{%
162   \ifnumberingR
163     \global\numberingRfalse
164     \normal@pars
165     \ifnum\l@dnumppstartsR=0%
166       \led@err@NumberingWithoutPstart%
167     \fi%
168     \ifl@dpairing
169       \global\pst@rtedRfalse
170     \else
171       \ifx\insertlines@listR\empty\else
172         \global\noteschanged@true
173       \fi
174       \ifx\line@listR\empty\else
175         \global\noteschanged@true
176       \fi
177     \fi
178     \ifnoteschanged@

```

```

179     \led@mess@NotesChanged
180     \fi
181     \else
182     \led@err@NumberingNotStarted
183     \fi
184     \endgroup
185     \if@noeled@sec\else%
186     \immediate\closeout\eled@sectioningR@out%
187     \fi%
188   }
189
190 %

```

`\initnumbering@sectcountR` We do not want the right side section commands to be numbered after the left side ones, instead we want them numbered after which is typeset before the pages or columns environments. we switch the \LaTeX counter in `\numberingR`.

`\save@section@number`

`\set@sectcountR`

```

191 \newcounter{chapterR}
192 \newcounter{sectionR}
193 \newcounter{subsectionR}
194 \newcounter{subsubsectionR}
195
196 \newcount\old@chapter%
197 \newcount\old@section%
198 \newcount\old@subsection%
199 \newcount\old@subsubsection%
200 \newcommand{\save@section@number}{%
201   \ifdefined\c@chapter%
202     \global\old@chapter\value{chapter}%
203   \fi%
204   \global\old@section\value{section}%
205   \global\old@subsection\value{subsection}%
206   \global\old@subsubsection\value{subsubsection}%
207 }%
208 \newcommand{\initnumbering@sectcountR}{%
209   \ifdefined\c@chapter%
210     \setcounter{chapterR}{\old@chapter}%
211   \fi%
212   \setcounter{sectionR}{\old@section}%
213   \setcounter{subsectionR}{\old@subsection}%
214   \setcounter{subsubsectionR}{\old@subsubsection}%
215   \set@sectcountR%
216 }
217 \newcommand{\set@sectcountR}{%
218   \let\c@chapter\c@chapterR%
219   \let\c@section\c@sectionR%
220   \let\c@subsection\c@subsectionR%
221   \let\c@subsubsection\c@subsubsectionR%
222 }%
223 %

```

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

```

224 \newcommand*{\pausenumberingR}{%
225   \endnumberingR\global\numberingRtrue}
226 \newcommand*{\resumenumberingR}{%
227   \ifnumberingR
228     \global\pst@rtedRtrue
229     \global\advance\section@numR \@ne
230     \led@mess@SectionContinued{\the\section@numR R}%
231     \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
232     \l@dend@stuff
233     \begingroup%
234     \initnumbering@sectcountR%
235   \else
236     \led@err@numberingShouldHaveStarted
237     \endnumberingR
238     \beginnumberingR
239   \fi}
240
241 %

```

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

```

242 \newcommand*{\memorydumpL}{%
243   \endnumbering
244   \numberingtrue
245   \global\pst@rtedLtrue
246   \global\advance\section@num \@ne
247   \led@mess@SectionContinued{\the\section@num}%
248   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
249   \l@dend@stuff}
250
251 \newcommand*{\memorydumpR}{%
252   \endnumberingR
253   \numberingRtrue
254   \global\pst@rtedRtrue
255   \global\advance\section@numR \@ne
256   \led@mess@SectionContinued{\the\section@numR R}%
257   \line@list@stuffR{\jobname.\extensionchars\the\section@numR R}%
258   \l@dend@stuff}
259
260 %

```

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`.

```

261 \newcommand*{\lineationR}[1]{%
262   \ifnumbering
263     \led@err@LineationInNumbered
264   \else
265     \def\@tempa{#1}\def\@tempb{page}%
266     \ifx\@tempa\@tempb
267       \global\bypage@Rtrue
268       \global\bypstart@Rfalse
269       \unless\ifnocritical@%
270         \Xpstart[] [false]%
271       \fi%
272     \else
273       \def\@tempb{pstart}%
274       \ifx\@tempa\@tempb
275         \global\bypage@Rfalse
276         \global\bypstart@Rtrue
277         \unless\ifnocritical@%
278           \Xpstart%
279         \fi%
280       \else
281         \def\@tempb{section}
282         \ifx\@tempa\@tempb
283           \global\bypage@Rfalse%
284           \global\bypstart@Rfalse%
285           \unless\ifnocritical@%
286             \Xpstart[] [false]%
287           \fi%
288         \else
289           \led@warn@BadLineation
290         \fi%
291       \fi
292     \fi
293   \fi}}
294 %

```

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

```

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

```

```

296 \lineation{#1}%
297 \lineationR{#1}%
298 }%
299 %

```

IV.2 Setting line number margin

`\linenummargin` You call `\linenummargin{<word>}` to specify which margin you want your right text's line numbers in; it takes one argument, a string. You can put the line numbers in the same margin on every page using `left` or `right`; or you can use `inner` or `outer` to get them in the inner or outer margins. You can change this within a numbered section, but the change may not take effect just when you would like; if it is done between paragraphs nothing surprising should happen.

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

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

```

300 \newcount\line@marginR
301 %

```

By default put right text numbers at the right.

```

302 \line@marginR=\@ne
303
304 %

```

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

```

305 \newcommand{\linenummarginR}[1]{%
306   \l@getline@margin{#1}%
307   \ifnum\@l@tempcntb>\m@ne%
308     \global\line@marginR=\@l@tempcntb%
309   \fi%
310 }
311 \WithSuffix\newcommand\linenummargin*[1]{%
312   \l@getline@margin{#1}%
313   \ifnum\@l@tempcntb>\m@ne%
314     \global\line@marginR=\@l@tempcntb%
315     \global\line@margin=\@l@tempcntb%
316   \fi%
317 }
318 %

```

IV.3 Setting lineation start and step

`\c@firstlinenumR` The following counters tell `reledmac` which right text lines should be printed with line numbers. `firstlinenumR` is the number of the first line in each section that gets a number; `linenumincrementR` is the difference between successive numbered lines. The ini-

tial values of these counters produce labels on lines 5, 10, 15, etc. `linenumincrementR` must be at least 1.

```

319 \newcounter{firstlinenumR}
320 \setcounter{firstlinenumR}{5}
321 \newcounter{linenumincrementR}
322 \setcounter{linenumincrementR}{5}
323 %

```

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

```

324 \newcounter{firstsublinenumR}
325 \setcounter{firstsublinenumR}{5}
326 \newcounter{sublinenumincrementR}
327 \setcounter{sublinenumincrementR}{5}
328
329 %

```

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

```

330 \firstsublinenum \WithSuffix\newcommand\firstlinenum*[1]{%
331 \sublinenumincrement \setcounter{firstlinenumR}{#1}%
332 \firstlinenum* \setcounter{firstlinenum}{#1}%
333 \linenumincrement* }
334 \firstsublinenum* \WithSuffix\newcommand\linenumincrement*[1]{%
335 \sublinenumincrement* \setcounter{linenumincrementR}{#1}%
336 \setcounter{linenumincrement}{#1}%
337 }
338 \WithSuffix\newcommand\firstsublinenum*[1]{%
339 \setcounter{subfirstlinenumR}{#1}%
340 \setcounter{subfirstlinenum}{#1}%
341 }
342 \WithSuffix\newcommand\sublinenumincrement*[1]{%
343 \setcounter{sublinenumincrementR}{#1}%
344 \setcounter{sublinenumincrement}{#1}%
345 }
346 %

```

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

```

347 \linenumincrementR \newcommand\firstlinenumR[1]{%
348 \firstsublinenumR \setcounter{firstlinenumR}{#1}%
349 \sublinenumincrementR }
350 \newcommand\linenumincrementR[1]{%
351 \setcounter{linenumincrementR}{#1}%
352 }
353 \newcommand\firstsublinenumR[1]{%
354 \setcounter{subfirstlinenumR}{#1}%

```



```

355 }
356 \newcommand\sublinenumincrementR[1]{%
357   \setcounter{sublinenumincrementR}{#1}%
358 }
359 %

```

IV.4 Setting line flag

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

```

360 \newcommand{\setRlineflag}[1]{%
361   \gdef\Rlineflag{#1}%
362 }
363 \setRlineflag{R}
364 %

```

IV.5 Setting line number style

\linenumrepR **\sublinenumrepR** $\langle ctr \rangle$ typesets the right line number $\langle ctr \rangle$, and similarly **\sublinenumrepR** for subline numbers.

```

365 \newcommand*\linenumrepR[1]{\@arabic{#1}}
366 \newcommand*\sublinenumrepR[1]{\@arabic{#1}}
367
368 %

```

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

```

369 \newcommand*\linenumberstyleR[1]{%
370   \def\linenumrepR##1{\@nameuse{@#1}{##1}}
371 \newcommand*\sublinenumberstyleR[1]{%
372   \def\sublinenumrepR##1{\@nameuse{@#1}{##1}}
373 %

```

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

```

374 \WithSuffix\newcommand\linenumberstyle*[1]{%
375   \linenumberstyle{#1}%
376   \linenumberstyleR{#1}%
377 }%
378
379 \WithSuffix\newcommand\sublinenumberstyle*[1]{%
380   \sublinenumberstyle{#1}%
381   \sublinenumberstyleR{#1}%
382 }%
383 %
384 %

```

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`.

```

385 \newcommand*{\leftlinenumR}{%
386   \l@dlinenumR
387   \kern\linenumsep}
388 \newcommand*{\rightlinenumR}{%
389   \kern\linenumsep
390   \l@dlinenumR}
391 \newcommand*{\l@dlinenumR}{%
392   \numlabfont\linenumrepR{\line@numR}\@Rlineflag%
393   \ifsublines@
394     \ifnum\subline@num>\z@
395       \unskip\fullstop\sublinenumrepR{\subline@numR}%
396     \fi
397   \fi}
398
399 %

```

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.

```

400 \newcount\line@numR
401 \newcount\subline@numR
402 \newcount\absline@numR
403
404 %

```

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

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

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

```

405 \list@create{\line@listR}
406 \list@create{\insertlines@listR}
407 \list@create{\actionlines@listR}
408 \list@create{\actions@listR}

```

```
409
410 %
```

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

```
411 \newcount\page@numR
412
413 %
```

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`

```
414 \list@create{\linesinpar@listL}
415 \list@create{\linesinpar@listR}
416 \list@create{\maxlinesinpar@list}
417
418 %
```

IV.8 Reading the line-list file

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

```
419 \newcommand{\list@clearing@regR}{%
420   \list@clear{\line@listR}%
421   \list@clear{\insertlines@listR}%
422   \list@clear{\actionlines@listR}%
423   \list@clear{\actions@listR}%
424   \list@clear{\linesinpar@listR}%
425   \list@clear{\linesonpage@listR}
426 }
427 %
```

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

```
428 \newcommand{\@par@sync@option}[1]{%
429   \IfStrEq{#1}{\@par@this@sync@option}%
430   {}%
431   {\ifledRcol%
432     \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
433   }%
434   \else%
435     \led@warn@ChangeSyncOption{\jobname.\extensionchars\the\section@num}
436   }%
437   \fi%
438   \endinput%
```

```

437 }%
438 }%
439 %

```

`\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.

```

440 \newcommand{\@nl@regR}{%
441   \ifx\l@dchset@num\relax \else
442     \advance\absline@numR \@ne
443     \set@line@action
444     \let\l@dchset@num\relax
445     \advance\absline@numR \m@ne
446     \advance\line@numR \m@ne%    % do we need this?
447   \fi
448   \advance\absline@numR \@ne
449   \ifx\next@page@numR\relax \else
450     \page@action
451     \let\next@page@numR\relax
452   \fi
453   \ifx\sub@change\relax \else
454     \ifnum\sub@change>\z@
455       \sublines@true
456     \else
457       \sublines@false
458     \fi
459     \sub@action
460     \let\sub@change\relax
461   \fi
462   \ifcase\@lockR
463   \or
464     \@lockR \tw@
465   \or\or
466     \@lockR \z@
467   \fi
468   \ifcase\sub@lockR
469   \or

```

```

470 \sub@lockR \tw@
471 \or\or
472 \sub@lockR \z@
473 \fi
474 \ifsublines@
475 \ifnum\sub@lockR<\tw@
476 \advance\subline@numR \@ne
477 \fi
478 \else
479 \ifnum\@lockR<\tw@
480 \advance\line@numR \@ne \subline@numR \z@
481 \fi
482 \fi}
483
484
485 %

```

`\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`.

```

486 \newcount\last@page@numR
487 \last@page@numR=-10000
488
489 %

```

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

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

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

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

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

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

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

```

490 \newcount\@lockR
491 \newcount\sub@lockR
492
493 \newcommand*\do@lockonR}{%
494   \xright@appenditem{\the\absline@numR}\to\actionlines@listR
495   \ifsublines@
496     \xright@appenditem{-1005}\to\actions@listR
497     \ifnum\sub@lockR=\z@
498       \sub@lockR \@ne
499     \else
500       \ifnum\sub@lockR=\thr@@
501         \sub@lockR \@ne
502       \fi
503     \fi
504   \else
505     \xright@appenditem{-1003}\to\actions@listR
506     \ifnum\@lockR=\z@
507       \@lockR \@ne
508     \else
509       \ifnum\@lockR=\thr@@
510         \@lockR \@ne
511       \fi
512     \fi
513   \fi}
514
515 %

```

`\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@lockoff` `\do@lockoffR`

`\skip@lockoff`

```

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

```

```

535
536
537 %

```

`\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`.

```

538 \newcount\insert@countR
539 %

```

- #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.

```

540 \newcommand*{\@ref@regR}[2]{%
541   \global\advance\@edtext@level by 1%
542   \global\insert@countR=#1\relax
543   \loop\ifnum\insert@countR>\z@
544     \xright@appenditem{\the\absline@numR}\to\insertlines@listR
545     \global\advance\insert@countR \m@ne
546   \repeat
547 %

```

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.

```

548 \begingroup
549   \let\@ref=\dummy@ref
550   \let\@lopR\@gobble
551   \let\page@action=\relax
552   \let\sub@action=\relax
553   \let\set@line@action=\relax
554   \let\@lab=\relax
555   \let\@lemma=\relax
556   \let\@sw\@gobblethree%
557   #2

```

```

558 \global\endpage@num=\page@numR
559 \global\endline@num=\line@numR
560 \global\endsubline@num=\subline@numR
561 \endgroup
562 %

```

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

```

563 \xright@appenditem%
564   {\the\page@numR|\the\line@numR|}%
565   \ifsublines@ \the\subline@numR \else 0\fi|}%
566   \the\endpage@num|\the\endline@num|}%
567   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@listR
568 %

```

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

```

569 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
@edtext@level\endcsname}%
570 %

```

Declare and init boolean for lemma in this level.

```

571 \providebool{lemmacommand@\the\@edtext@level}%
572 \boolfalse{lemmacommand@\the\@edtext@level}%
573 %

```

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

```

574 #2
575 % Now, we store the list of \protect\cs{@sw} of this current \protect\cs{
edtext} as an element of
576 % the global list of list of \protect\cs{@sw} for a \protect\cs{edtext}
depth.
577 % \begin{macrocode}
578 \ifnum\@edtext@level>0%
579 \def\create@this@edtext@level{\expandafter\list@create\expandafter{
csname sw@list@edtextR@\the\@edtext@level\endcsname}}%
580 \ifcsundef{sw@list@edtextR@\the\@edtext@level}{\
create@this@edtext@level}{}%
581 \letcs{\@tmp}{sw@list@edtextR@\the\@edtext@level}%
582 \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}%
583 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
584 \global\cslet{sw@list@edtextR@\the\@edtext@level}{\@tmp}%
585 \fi%
586 %

```

Decrease edtext level counter.

```

587 \global\advance\@edtext@level by -1%
588 }
589 %

```


\@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.

```

590 \renewcommand*{\@pend}[1]{%
591   \ifbypstart@global\line@num=0\fi%
592   \xright@appenditem{#1}\to\linesinpar@listL}
593 \renewcommand*{\@pendR}[1]{%
594   \ifbypstart@R\global\line@numR=0\fi
595   \xright@appenditem{#1}\to\linesinpar@listR}
596
597 %

```

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

```

598 \newcommand{\@pstart}[3]{%
599   \ifcsdef{minpage@pstart@#1}%
600     {\ifnumgreater{#2}{\csuse{minpage@pstart@#1}}%
601       {\csnumgdef{minpage@pstart@#1}{#2}}%
602       {}%
603     }%
604     {\csnumgdef{minpage@pstart@#1}{#2}}
605   \csnumgdef{afterlines@pstart@#1L}{#3}%
606   }%
607
608 \newcommand{\@pstartR}[3]{%
609   \numdef{\@tmp}{#2-1}%Because we have not to know in which page the pstart
610   \ifcsdef{minpage@pstart@#1}%
611     {\ifnumgreater{\@tmp}{\csuse{minpage@pstart@#1}}%
612       {\csnumgdef{minpage@pstart@#1}{\@tmp}}%
613       {}%
614     }%
615     {\csnumgdef{minpage@pstart@#1}{\@tmp}}
616   \csnumgdef{afterlines@pstart@#1R}{#3}%
617   }%
618 %

```

\@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.

```

619 \renewcommand*{\@lopL}[1]{%
620   \xright@appenditem{#1}\to\linesonpage@listL}
621 \renewcommand*{\@lopR}[1]{%
622   \xright@appenditem{#1}\to\linesonpage@listR}
623
624 %

```

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`.

```
625 \newwrite\linenum@outR
626 %
```

`\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
627 \newif\iffirst@linenum@out@R
628 \first@linenum@out@Rtrue
629 %
```

`\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.

```
630 \newcommand*{\line@list@stuffR}[1]{%
631   \read@linelist{#1}%
632   \iffirst@linenum@out@R
633     \immediate\closeout\linenum@outR
634     \global\first@linenum@out@Rfalse
635     \immediate\openout\linenum@outR=#1
636     \immediate\write\linenum@outR{\string\line@list@version{\
this@line@list@version}}}%
637     \ifl@dpaging%
638       \immediate\write\linenum@outR{\string\@par@sync@option{\
@par@this@sync@option}}}%
639     \fi%
640   \else
641     \if@minipage%
642       \leavevmode%
643     \fi%
644     \closeout\linenum@outR%
645     \openout\linenum@outR=#1%
646   \fi}
647
648 %
```

`\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.

```
649 \newcommand*{\new@lineL}{%
650   \write\linenum@out{\string\@nl[\the\c@page][\thepage]]}
651 %
```

`\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.

```

652 \newcommand*{\new@lineR}{\%
653   \write\linenum@outR{\string\@nl[\the\c@page][\thepage]}}
654 \%

```

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

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

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

`\setline` You can use `\setline{<num>}` in running text (i.e., within `\pstart... \pend`) to set the current visible line-number to a specified positive value. It is defined in `reledmac`.

`\setlinenum` You can use `\setlinenum{<num>}` before a `\pstart` to set the visible line-number to a specified positive value. It writes a `\l@d@set` command to the line-list file. It is defined in `reledmac`.

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

`\skipnumbering`

V Marking text for notes

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

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

V.1 Specific hooks and commands for notes

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

```
\newseries@par 655 \newcommand{\newseries@par}[1]{%
656 %
```

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.

```
657 \unless\ifnofamiliar@%
658 \csgdef{onlysideX@#1}{}%
659 \global\newbool{keepforsideX@#1}%
660 \fi%
661 \unless\ifnocritical@%
662 \global\newbool{keepforXside@#1}%
663 \csgdef{Xonlyside@#1}{}%
664 \fi%
665 %
```

V.2 Tools specific to familiar footnotes

```
666 \unless\ifnofamiliar@%
667 %
```

V.2.1 Managing correct number

One problem with using familiar footnotes in parallel typesetting is the fact that the order of reading notes is not the same as the order they are typeset, because \LaTeX reads first all the notes on one side, then all the notes on the other side. Then, however, \LaTeX alternates between typesetting left-side note and right-side notes. Consequently, if we do nothing special, the note numbers are sorted in the reading order, not in the typesetting order. So we could obtain something like 1,3,2,5,4.

To prevent this problem, we use a two new counters by series. Every note, in parallel typesetting, has three associated counters.

1. A \LaTeX counter `footnoteX`. This the only one manipulated by user, and the only one finally printed.
2. A \TeX counter `footnoteX@reading`. Its value is incremented when reading the `\footnoteX` command in left or right side environments. It is used to get the correct footnote number from the `.aux` file to be typeset in the main text.
3. A \LaTeX counter `footnoteX@typeset`. Its value is increased when inserting footnotes. Its files to be used on the next run for the main text.

So here, we only defined the new counters.

```
668 \expandafter\newcount\csname footnote#1@reading\endcsname%
669 \newcounter{footnote#1@typeset}%
670 %
```

V.2.2 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`.

So declare the list.

```
671 \expandafter\list@create\csname footnote#1@mk\endcsname%
672 %
```

Then, declare the `\footnoteXnomk` command.

```
673 \expandafter\newcommand\csname footnote#1nomk\endcsname[1]{%
674 %
```

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

```
675 \toggletrue{nomk@}%
676 \csuse{footnote#1}{##1}%
677 \togglefalse{nomk@}%
678 %
```

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`.

```
679 \letcs{\@tmp}{footnote#1@mk}%
680 \numdef\@tmpa{\csuse{c@footnote#1}}%
681 \global\xright@appenditem{\@tmpa}\to\@tmp%
682 \global\cslet{footnote#1@mk}{\@tmp}%
683 }%
684 %
```

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

```
685 \expandafter\newcommand\csname footnote#1mk\endcsname{%
686 %
```

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

```
687 \letcs{\@tmp}{footnote#1@mk}%
688 \gl@p\@tmp\to\@tmpa%
689 \global\cslet{footnote#1@mk}{\@tmp}%
690 %
```

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

```
691 \letcs{\old@footnote}{c@footnote#1}%
692 \setcounter{footnote#1}{\@tmpa}%
693 %
```

Define the footnote mark and print it

```

694 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
695 \csuse{@footnotemark#1}%
696 %

```

Restore previous footnote counter and finally add space.

```

697 \setcounter{footnote#1}{\old@footnote}%
698 \xspace%
699 }%
700 %

```

End of tools specific to familiar notes.

```

701 \fi
702 %

```

End of \newseries@par.

```

703 }%
704 %

```

V.2.3 Get correct footnote number

`\get@familiarfootnote@number` As users can insert footnotes between two `\Pairs` or `\Pages` commands, we have to
`\save@familiarfootnote@number` set the `\+footnoteX@typeset+` counter to the last value of the `footnoteX` counter at
the beginning of these two commands.

```

705 \newcommand{\save@familiarfootnote@number}{%
706 \unless\ifnofamiliar@%
707 \def\do##1{\csxdef{saved@footnote##1}{\the\csname c@footnote##1\
endcsname}}%
708 \dolistloop{\@series}%
709 \fi%
710 \xdef\saved@footnote{\the\c@footnote}%
711 }
712 \newcommand{\get@familiarfootnote@number}{%
713 \unless\ifnofamiliar@%
714 \def\do##1{\setcounter{footnote##1@typeset}{\csuse{saved@footnote##1}}%
715 %
716 \dolistloop{\@series}%
717 \fi%
718 \setcounter{footnote@typeset}{\saved@footnote}%
719 }

```

V.3 Create hooks

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

```

720 \unless\ifnocritical@%
721   \newhookcommand@series{Xonlyside}%
722 \fi%
723 \unless\ifnofamiliar@%
724   \newhookcommand@series{onlysideX}%
725 \fi
726
727
728 %

```

V.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`.

```

729 \newcommand{\init@series@par}{%
730   \def\do##1{\newseries@par{##1}}%
731   \dolistloop{\@series}%
732 }%
733 \init@series@par%
734 %

```

V.5 Tools specific to \LaTeX 's classical footnotes

As users can use classical footnotes of \LaTeX (`\footnote`) in parallel texts, we must integrate the same tools to get correct number as for `reledmac`' footnotes (V.2.1 p. 44).

```

\footnote@reading35 \newcount\footnote@reading%
\footnote@typeset36 \newcounter{footnote@typeset}%
737 %

```

VI Pstart numbers dumping and restoration

While in `reledmac` the footnotes are inserted at 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@pc38 \list@create{\list@pstartL@pc}%
\list@pstartR@pc39 \list@create{\list@pstartR@pc}%
740 %

```

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 41 \def\dump@pstartL@pc{%
\dump@pstartR@pc 42 \xright@appenditem{\the\c@pstartL}\to\list@pstartL@pc%
743 \global\cslet{numberpstartL\the\l@dumpstartsL}{\ifnumberpstart}%
744 }%
745
746 \def\dump@pstartR@pc{%
747 \xright@appenditem{\the\c@pstartR}\to\list@pstartR@pc%
748 \global\cslet{numberpstartR\the\l@dumpstartsR}{\ifnumberpstart}%
749 }%
750
751 %

```

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

```

\restore@pstartR@pc
752 \def\restore@pstartL@pc{%
753 \ifx\list@pstartL@pc\empty\else%
754 \gl@p\list@pstartL@pc\to\@temp%
755 \global\c@pstartL=\@temp%
756 \fi%
757 }%
758 \def\restore@pstartR@pc{%
759 \ifx\list@pstartR@pc\empty\else%
760 \gl@p\list@pstartR@pc\to\@temp%
761 \global\c@pstartR=\@temp%
762 \fi%
763 }%
764 %

```

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.

```

765 \newenvironment{pairs}{%}
766 \l@dpairingtrue
767 \l@dpagingfalse
768 \initnumbering@quote
769 \save@familiarfootnote@number%
770 \save@section@number%
771 \at@begin@pairs%
772 }{%

```



```

773 \l@dpairingfalse
774 }
775
776 %

```

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

```

777 \newcommand{\AtBeginPairs}[1]{\xdef\at@begin@pairs{#1}}%
778 \def\at@begin@pairs{}%
779
780 %

```

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.

```

781 \newenvironment{pages}{%
782 \l@dpairingtrue
783 \l@dpagingtrue
784 \initnumbering@quote
785 \save@familiarfootnote@number%
786 \save@section@number%
787 \setlength{\Lcolwidth}{\textwidth}%
788 \setlength{\Rcolwidth}{\textwidth}%
789 }{%
790 \l@dpairingfalse
791 \l@dpagingfalse
792 }
793
794 %

```

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

```

795 \newif\ifinstanzaL
796 \newif\ifinstanzaR
797 %

```

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.

```

798 \newenvironment{Leftside}{%
799 \expandafter\ifvoid\csname l@dLcolrawbox1\endcsname\else%
800 \led@err@Leftside@PreviousNotPrinted%
801 \fi%
802 \ledRcolfalse

```

```

803 \setcounter{pstartL}{1}
804 \let\pstart\pstartL
805 \let\thepstart\thepstartL
806 \let\pend\pendL
807 \let\memorydump\memorydumpL
808 \Leftsidehook
809 \let\old@startstanza\@startstanza
810 \def\@startstanza[##1]{\global\instanzaLtrue\old@startstanza[##1]}
811 }{
812 \Leftsidehookend}
813 %

```

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

```

\Leftsidehookend
\Rightsidehook
\Rightsidehookend
814 \newcommand*\Leftsidehook{}
815 \newcommand*\Leftsidehookend{}
816 \newcommand*\Rightsidehook{}
817 \newcommand*\Rightsidehookend{}
818
819 %

```

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.

```

820 \newenvironment{Rightside}{%
821 \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname\else%
822 \led@err@Rightside@PreviousNotPrinted%
823 \fi%
824 \ledRcoltrue
825 \let\beginnumbering\beginnumberingR
826 \let\endnumbering\endnumberingR
827 \let\pausenumbering\pausenumberingR
828 \let\resumenumbering\resumenumberingR
829 \let\memorydump\memorydumpR
830 \let\thepstart\thepstartR
831 \let\pstart\pstartR
832 \let\pend\pendR
833 \let\ledpb\ledpbR
834 \let\lednopb\lednopbR
835 \let\lineation\lineationR
836 \Rightsidehook
837 \let\old@startstanza\@startstanza
838 \def\@startstanza[##1]{\global\instanzaRtrue\old@startstanza[##1]}
839 }{%
840 \ledRcolfalse
841 \Rightsidehookend
842 }
843

```

844 %

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.

```
845 \newcount\num@linesR
846 \newbox\one@lineR
847 \newcount\par@lineR
848 %
```

`\pstartL` `\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant variables, and then arranges for the subsequent text to go into the appropriate box. `\pstart` needs to appear at the start of every paragraph that is to be numbered.

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.

```
849
850 \newcounter{pstartL}
851 \renewcommand{\thepstartL}{\bfseries\@arabic\c@pstartL}. }
852 \newcounter{pstartR}
853 \renewcommand{\thepstartR}{\bfseries\@arabic\c@pstartR}. }
854
855 \newcommandx*\pstartL[1][1]{%
856   \if@nobreak%
857     \let\@oldnobreak\@nobreaktrue%
858   \else%
859     \let\@oldnobreak\@nobreakfalse%
```

```

860 \fi%
861 \@nbreaktrue%
862 \ifluatex%
863 \xdef\l@luatextextdir@L{\the\textdir}%
864 \xdef\l@luatexpardir@L{\the\pardir}%
865 \xdef\l@luatexbodydir@L{\the\bodydir}%
866 \fi%
867 \ifnumbering \else%
868 \led@err@PstartNotNumbered%
869 \beginnumbering%
870 \fi%
871 \ifnumberedpar@%
872 \led@err@PstartInPstart%
873 \pend%
874 \fi%
875 %

```

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

```

876 \ifpst@rtedL\else%
877 \list@clear{\inserts@list}%
878 \global\let\next@insert=\empty%
879 \global\pst@rtedLtrue%
880 \fi%
881 \begingroup\normal@pars%
882 %

```

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.

```

883 \global\advance\l@dnumpstartsL \@ne%
884 \ifnum\l@dnumpstartsL>\l@dc@maxchunks%
885 \led@err@TooManyPstarts%
886 \global\l@dnumpstartsL=\l@dc@maxchunks%
887 \fi%
888 \global\setnamebox{\l@dLcolrawbox\the\l@dnumpstartsL}=\vbox\bgroup%
889 %

```

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.

```

890 \l@dzeropenalties%
891 \ifautopar\else%
892 \ifnumberpstart%
893 \ifsidepstartnum%
894 \else%
895 \thepstartL%
896 \fi%
897 \fi%
898 \fi%

```

```

899 \hsize=\Lcolwidth%
900 \numberedpar@true%
901 \iflabelpstart\protected@edef\@currentlabel%
902     {\p@pstartL\thepstartL}\fi%
903 %

```

Dump the optional arguments

```

904 \ifstrempy{#1}%
905     {\csgdef{before@pstartL@the\l@dnumpstartsL}{\at@every@pstart}}%
906     {\csgdef{before@pstartL@the\l@dnumpstartsL}{\noindent#1}}%
907     \at@every@pstart@call%
908 %

```

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

```

909 \ignorespaces%
910 %

```

```

911 }
912 %

```

The same for right side.

```

913 \newcommandx*{\pstartR}[1][1]{%
914     \if@nbreak%
915         \let\@oldnbreak\@nbreaktrue%
916     \else%
917         \let\@oldnbreak\@nbreakfalse%
918     \fi%
919     \@nbreaktrue%
920     \ifluatex%
921         \xdef\l@luatextextdir@R{\the\textdir}%
922         \xdef\l@luatexpardir@R{\the\pardir}%
923         \xdef\l@luatexbodydir@R{\the\bodydir}%
924     \fi%
925     \ifnumberingR \else%
926         \led@err@PstartNotNumbered%
927         \beginnumberingR%
928     \fi%
929     \ifnumberedpar@%
930         \led@err@PstartInPstart%
931     \pendR%
932 \fi%
933 \ifpst@rtedR\else%
934     \list@clear{\inserts@listR}%
935     \global\let\next@insertR=\empty%
936     \global\pst@rtedRtrue%
937 \fi%
938 \begingroup\normal@pars%
939 \global\advance\l@dnumpstartsR \@one%
940 \ifnum\l@dnumpstartsR>\l@dc@maxchunks%

```

```

941 \led@err@TooManyPstarts%
942 \global\l@dnumpstartsR=\l@dc@maxchunks%
943 \fi%
944 \global\setnamebox{l@dRcolrawbox\the\l@dnumpstartsR}=\vbox\bgroup%
945 \l@dzeropenalties%
946 \ifautopar\else%
947 \ifnumberpstart%
948 \ifsidepstartnum\else%
949 \thepstartR%
950 \fi%
951 \fi%
952 \fi%
953 \hsize=\Rcolwidth%
954 \numberedpar@true%
955 \iflabelpstart\protected@edef\@currentlabel%
956 {\p@pstartR\thepstartR}\fi%
957 \ifstreempty{#1}%
958 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\at@every@pstart}}%
959 {\csgdef{before@pstartR@the\l@dnumpstartsR}{\noindent#1}}%
960 \at@every@pstart@call%
961 \ignorespaces%
962 }
963 %

```

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

```

964 \newcommandx*\pendL}[1][1]{%
965 \ifnumbering \else%
966 \led@err@PendNotNumbered%
967 \fi%
968 \ifnumberedpar@ \else%
969 \led@err@PendNoPstart%
970 \fi%
971 %

```

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.

```

972 \endgraf\global\num@lines=\prevgraf\egroup%
973 \global\par@line=0%
974 %

```

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

```

975 \endgroup%
976 \ignorespaces%
977 \@oldnobreak%
978 \dump@pstartL@pc%
979 \ifnumberpstart%
980 \addtocounter{pstartL}{1}%
981 \fi

```

```

982 \parledgroup@beforenotes@save{L}%
983 %
    Dump content of the optional argument.
984 \ifstrempy{#1}%
985   {\csgdef{after@pendL@the\l@dnumpstartsL}{\at@every@pend}}%
986   {\csgdef{after@pendL@the\l@dnumpstartsL}{\noindent#1}}%
987 }
988 %

```

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

```

989 \newcommandx*{\pendR}[1][1]{%
990   \ifnumberingR \else%
991     \led@err@PendNotNumbered%
992   \fi%
993   \ifnumberedpar@ \else%
994     \led@err@PendNoPstart%
995   \fi%
996   \endgraf\global\num@linesR=\prevgraf\egroup%
997   \global\par@lineR=0%
998   \endgroup%
999   \ignorespaces%
1000   \@oldnobreak%
1001   \dump@pstartR@pc%
1002   \ifnumberpstart%
1003     \addtocounter{pstartR}{1}%
1004   \fi%
1005   \parledgroup@beforenotes@save{R}%
1006   \ifstrempy{#1}%
1007     {\csgdef{after@pendR@the\l@dnumpstartsR}{\at@every@pend}}%
1008     {\csgdef{after@pendR@the\l@dnumpstartsR}{\noindent#1}}%
1009   }
1010 %
1011 %

```

`\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.

```

1012 \newcommand{\AtEveryPstartCall}[1]{\gdef\at@every@pstart@call{#1}}%
1013 \gdef\at@every@pstart@call{}%
1014 %

```

`\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`

```

1015 \newif\ifprint@last@after@pendL%
1016 \newif\ifprint@last@after@pendR%
1017 %

```

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`

```
1018 \newbox\l@dleftbox
1019 \newbox\l@drightbox
1020
1021 %
```

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

```
\countRline
1022 \newcount\countLline
1023 \countLline \z@
1024 \newcount\countRline
1025 \countRline \z@
1026
1027 %
```

`\@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`

```
1028 \newcount\@donereallinesL
1029 \newcount\@donetotallinesL
1030 \newcount\@donereallinesR
1031 \newcount\@donetotallinesR
1032
1033 %
```

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

```
1034 \newcommand*{\do@lineL}{%
1035 \letcs{\ifnumberpstart}{numberpstart@L\the\l@dpscl}%
1036 \advance\countLline \@ne%
1037 \ifvbox\namebox{\l@dLcolrawbox\the\l@dpscl}%
1038 {\vbadness=10000%
1039 \splittopskip=\z@%
1040 \do@lineLhook%
1041 \l@demtyd@ta%
1042 \global\setbox\one@line=\vsplit\namebox{\l@dLcolrawbox\the\l@dpscl}%
1043 to\baselineskip}%
1044 \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
parledgroup@notes@startL}{}%
1045 \unvbox\one@line \global\setbox\one@line=\lastbox%
1046 \@writepageofparL%
```



```

1047 \getline@numL%
1048 \ifnum\@lock>\@ne%
1049   \inserthangingsymboltrue%
1050 \else%
1051   \inserthangingsymbolfalse%
1052 \fi%
1053 \setbox\l@dleftbox%
1054 \hb@xt@ \Lcolwidth{%
1055   \ifl@dhiddenumber%
1056     \global\l@dhiddenumberfalse%
1057     \f@x@l@cks%
1058   \else%
1059     \affixline@num%
1060   \fi%
1061   \xifinlist{\the\l@dpscL}{\eled@sections@@}%
1062   {\add@inserts\affixside@note}%
1063   {\print@lineL}%
1064 }%
1065 \add@penaltiesL%
1066 \global\advance\@donereallinesL\@ne%
1067 \global\advance\@donetotallinesL\@ne%
1068 \else%
1069   \setbox\l@dleftbox \hb@xt@ \Lcolwidth{\hspace*{\Lcolwidth}}%
1070   \global\advance\@donetotallinesL\@ne%
1071 \fi%
1072 }%
1073
1074
1075 %

```

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

```

1076 \def\print@lineL{%
1077   \affixpstart@numL%
1078   \l@dld@ta %space kept for backward compatibility
1079   \add@inserts\affixside@note%
1080   \l@dlsn@te %space kept for backward compatibility
1081   \hb@xt@ \Lcolwidth{\ledllfill\hb@xt@ \wd\one@line{%
1082     \do@insidelineLhook%
1083     \ifluatex%
1084       \texdir\l@luatexttexdir@L%
1085     \fi%
1086     \new@lineL%
1087     \inserthangingsymbolL%
1088     \l@dunhbox@line{\one@line}}\ledrlfill\l@dldr@ta%
1089   \l@drrsn@te}}
1090
1091 %

```

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

```

1092 \def\print@eledsectionL{%
1093   \addtocounter{pstartL}{-1}%
1094   \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{%
1095     \ifdefstring{\@eledsectmark}{L}{\ledsectnomark}%
1096     \numdef{\temp@}{\l@dpscL-1}%
1097     \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1098     \@eled@sectioningtrue%
1099     \bgroup%
1100       \ifluatex%
1101         \texdir\l@luatextextdir@L%
1102         \pardir\l@luatexpardir@L%
1103         \bodydir\l@luatexbodydir@L%
1104         \ifdefstring{\l@luatextextdir@L}{TRT}{\@RTLtrue}{}%
1105       \fi%
1106       \csuse{eled@sectioning@the\l@dpscL}%
1107     \egroup%
1108     \@eled@sectioningfalse%
1109     \global\csundef{eled@sectioning@the\l@dpscL}%
1110     \if@RTL%
1111       \hspace{-3\paperwidth}%
1112       {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1113     \else%
1114       \hspace{3\paperwidth}%
1115       {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1116     \fi%
1117     \vskip\eledsection@correcting@skip%
1118   }
1119
1120 %

```

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

```

\doininsidelineLhook 1121 \newcommand*{\dolineLhook}[1]{\gdef\dolineLhook{#1}}%
\doininsidelineRhook 1122 \newcommand*{\dolineRhook}[1]{\gdef\dolineRhook{#1}}%
1123 \newcommand*{\doininsidelineLhook}[1]{\gdef\doininsidelineLhook{#1}}%
1124 \newcommand*{\doininsidelineRhook}[1]{\gdef\doininsidelineRhook{#1}}%
1125
1126 %

```

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

```

\dolineRhook 1127 \newcommand*{\do@lineLhook}{%
\doininsidelineLhook 1128 \newcommand*{\do@lineRhook}{%
\doininsidelineRhook 1129 \newcommand*{\do@insidelineLhook}{%
1130 \newcommand*{\do@insidelineRhook}{%
1131
1132 %

```

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

```

1133 \newcommand*{\do@lineR}{%
1134   \let\linenumrep\linenumrepR%
1135   \let\sublinenumrep\sublinenumrepR%
1136   \letcs{\ifnumberpstart}{numberpstart@R\the\l@dpscR}%
1137   \ledRcol@true%
1138   \advance\countRline \@ne%
1139   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}%
1140     {\vbadness=10000%
1141      \splittopskip=\z@%
1142      \do@lineRhook%
1143      \l@demptyd@ta%
1144      \global\setbox\one@lineR=\vsplit\namebox{1@dRcolrawbox\the\l@dpscR}%
1145        to\baselineskip}%
1146     \IfStrEq{\splitfirstmarks\parledgroup@}{begin}{\
parledgroup@notes@startR}{}%
1147     \unvbox\one@lineR \global\setbox\one@lineR=\lastbox%
1148     \@writepageofparR%
1149     \getline@numR%
1150     \ifnum\@lockR>\@ne%
1151       \inserthangingsymbolRtrue%
1152     \else%
1153       \inserthangingsymbolRfalse%
1154     \fi%
1155     \setbox\l@drightbox%
1156     \hb@xt@ \Rcolwidth{%
1157       \ifl@dhiddenumber%
1158         \global\l@dhiddenumberfalse%
1159         \f@x@l@cksR%
1160       \else%
1161         \affixline@numR%
1162       \fi%
1163       \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
1164       {\add@insertsR\affixside@noteR}%
1165       {\print@lineR}%
1166     }%
1167     \add@penaltiesR%
1168     \global\advance\@donereallinesR\@ne%
1169     \global\advance\@donetotallinesR\@ne%
1170   \else%
1171     \setbox\l@drightbox \hb@xt@ \Rcolwidth{\hspace*{\Rcolwidth}}%
1172     \global\advance\@donetotallinesR\@ne%
1173   \fi%
1174   \ledRcol@false%
1175 }
1176
1177
1178 %

```

`\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.

```

1179 \newcommand*{\getline@numR}{%
1180   \global\advance\absline@numR \@ne
1181   \do@actionsR
1182   \do@ballastR
1183   \ifledgroupnotesR\else
1184     \ifnumberline
1185       \ifsublines@
1186         \ifnum\sub@lockR<\tw@
1187           \global\advance\subline@numR \@ne
1188         \fi
1189       \else
1190         \ifnum\@lockR<\tw@
1191           \global\advance\line@numR \@ne
1192           \global\subline@numR \z@
1193         \fi
1194       \fi
1195     \fi
1196   \fi
1197 }
1198 \newcommand*{\getline@numL}{%
1199   \global\advance\absline@num \@ne
1200   \do@actions
1201   \do@ballast
1202   \ifledgroupnotesL\else
1203     \ifnumberline
1204       \ifsublines@
1205         \ifnum\sub@lock<\tw@
1206           \global\advance\subline@num \@ne
1207         \fi
1208       \else
1209         \ifnum\@lock<\tw@
1210           \global\advance\line@num \@ne
1211           \global\subline@num \z@
1212         \fi
1213       \fi
1214     \fi
1215   \fi
1216 }
1217
1218 %
1219 %

```

`\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.

```

1220 \newcommand*{\do@ballastR}{\global\ballast@count=\z@
1221   \begingroup
1222     \advance\absline@numR \@ne
1223     \ifnum\next@actionlineR=\absline@numR
1224       \ifnum\next@actionR>-1001
1225         \global\advance\ballast@count by -\c@ballast
1226       \fi
1227     \fi
1228   \endgroup}
1229 %

```

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

```

1230
1231 \newif\ifl@dskipversenumberR
1232 \newcommand*{\do@actions@fixedcodeR}{%
1233   \ifcase\@l@dttempcnta%
1234     \or% % 1001
1235       \global\sublines@true
1236     \or% % 1002
1237       \global\sublines@false
1238     \or% % 1003
1239       \global\@lockR=\@ne
1240     \or% % 1004%
1241       \ifnum\@lockR=\tw@
1242         \global\@lockR=\thr@@
1243       \else
1244         \global\@lockR=\z@
1245       \fi
1246     \or% % 1005
1247       \global\sub@lockR=\@ne
1248     \or% % 1006
1249       \ifnum\sub@lockR=\tw@
1250         \global\sub@lockR=\thr@@
1251       \else
1252         \global\sub@lockR=\z@
1253       \fi
1254     \or% % 1007
1255       \l@dskipnumbertrue
1256     \or% % 1008
1257       \l@dskipversenumberRtrue%
1258     \or% % 1009
1259       \l@dhidnumbertrue%
1260   \else%

```

```

1261 \led@warn@BadAction
1262 \fi%
1263 }
1264
1265
1266 \newcommand*{\do@actionsR}{%
1267 \global\let\do@actions@nextR=\relax
1268 \@l@dttempcntb=\absline@numR
1269 \ifnum\@l@dttempcntb<\next@actionlineR\else
1270 \ifnum\next@actionR>-1001\relax
1271 \global\page@numR=\next@actionR
1272 \ifbypage@R
1273 \global\line@numR \z@ \global\subline@numR \z@
1274 \fi
1275 \else
1276 \ifnum\next@actionR<-4999\relax % 9/05 added relax here
1277 \@l@dttempcnta=-\next@actionR
1278 \advance\@l@dttempcnta by -5001\relax
1279 \ifsublines@
1280 \global\subline@numR=\@l@dttempcnta
1281 \else
1282 \global\line@numR=\@l@dttempcnta
1283 \fi
1284 \else
1285 \@l@dttempcnta=-\next@actionR
1286 \advance\@l@dttempcnta by -1000\relax
1287 \do@actions@fixedcodeR
1288 \fi
1289 \fi
1290 \ifx\actionlines@listR\empty
1291 \gdef\next@actionlineR{1000000}%
1292 \else
1293 \glp\actionlines@listR\to\next@actionlineR
1294 \glp\actions@listR\to\next@actionR
1295 \global\let\do@actions@nextR=\do@actionsR
1296 \fi
1297 \fi
1298 \do@actions@nextR}
1299
1300 %

```

VIII.4 Line number printing

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

```

\ch@cksub@l@ckR
\ch@ck@l@ckR301
\fx@l@ckR302 \newcommand*{\l@dcalcn}{3}{%
\affixline@numR303 \ifnum #1 > #2\relax
304 \@l@dttempcnta = #1\relax

```

```

1305 \advance\@l@tempcnta by -#2\relax
1306 \divide\@l@tempcnta by #3\relax
1307 \multiply\@l@tempcnta by #3\relax
1308 \advance\@l@tempcnta by #2\relax
1309 \else
1310 \@l@tempcnta=#2\relax
1311 \fi}
1312
1313 \newcommand*{\ch@cksub@l@ckR}{%
1314 \ifcase\sub@lockR
1315 \or
1316 \ifnum\sublock@disp=\@ne
1317 \@l@tempcntb \z@ \@l@tempcnta \@ne
1318 \fi
1319 \or
1320 \ifnum\sublock@disp=\tw@
1321 \else
1322 \@l@tempcntb \z@ \@l@tempcnta \@ne
1323 \fi
1324 \or
1325 \ifnum\sublock@disp=\z@
1326 \@l@tempcntb \z@ \@l@tempcnta \@ne
1327 \fi
1328 \fi}
1329
1330 \newcommand*{\ch@ck@l@ckR}{%
1331 \ifcase\@lockR
1332 \or
1333 \ifnum\lock@disp=\@ne
1334 \@l@tempcntb \z@ \@l@tempcnta \@ne
1335 \fi
1336 \or
1337 \ifnum\lock@disp=\tw@
1338 \else
1339 \@l@tempcntb \z@ \@l@tempcnta \@ne
1340 \fi
1341 \or
1342 \ifnum\lock@disp=\z@
1343 \@l@tempcntb \z@ \@l@tempcnta \@ne
1344 \fi
1345 \fi}
1346
1347 \newcommand*{\f@x@l@cksR}{%
1348 \ifcase\@lockR
1349 \or
1350 \global\@lockR \tw@
1351 \or \or
1352 \global\@lockR \z@
1353 \fi
1354 \ifcase\sub@lockR

```

```

1355 \or
1356 \global\sub@lockR \tw@
1357 \or \or
1358 \global\sub@lockR \z@
1359 \fi}
1360
1361
1362 \newcommand*{\affixline@numR}{%
1363 \ifledgroupnotesR@ \else \ifnumberline
1364 \ifl@dskipnumber
1365 \global\l@dskipnumberfalse
1366 \else
1367 \ifsublines@
1368 \@l@tempcntb=\subline@numR
1369 \l@dcalcnun{\subline@numR}{\c@firstsublinenumR}{\c@sublinenumincrementR
}%
1370 \ch@cksub@lockR
1371 \else
1372 \@l@tempcntb=\line@numR
1373 \ifx\linenumberlist\empty
1374 \l@dcalcnun{\line@numR}{\c@firstlinenumR}{\c@linenumincrementR}%
1375 \else
1376 \@l@tempcnta=\line@numR
1377 \edef\rem@inder{\linenumberlist,\number\line@numR,%
1378 \edef\sc@n@list{\def\noexpand\sc@n@list
1379 ###1,\number\@l@tempcnta,###2|{\def\noexpand\rem@inder{###2}}}%
1380 \sc@n@list\expandafter\sc@n@list\rem@inder|}%
1381 \ifx\rem@inder\empty\advance\@l@tempcnta\@ne\fi
1382 \fi
1383 \ch@ck@l@ckR
1384 \fi
1385 \ifnum\@l@tempcnta=\@l@tempcntb
1386 \ifl@dskipversenumberR\else
1387 \if@twocolumn
1388 \if@firstcolumn
1389 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1390 \else
1391 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1392 \fi
1393 \else
1394 \@l@tempcntb=\line@marginR
1395 \ifnum\@l@tempcntb>\@ne
1396 \advance\@l@tempcntb by\page@numR
1397 \fi
1398 \ifodd\@l@tempcntb
1399 \gdef\l@drd@ta{\rlap{\rightlinenumR}}}%
1400 \else
1401 \gdef\l@dld@ta{\llap{\leftlinenumR}}}%
1402 \fi
1403 \fi

```



```

1404 \fi
1405 \fi
1406 \f@x@l@cksR
1407 \fi
1408 \fi
1409 \fi}
1410 %

```

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@numL11
\affixpstart@numR12 \newcommand*{\affixpstart@numL}{%
\leftpstartnumR13 \ifsidepstartnum
\rightpstartnumR14 \if@twocolumn
\leftpstartnumR15 \if@firstcolumn
\rightpstartnumR16 \gdef\l@dld@ta{\llap{\leftpstartnumL}}%
\ifpstartnumR17 \else
1418 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}%
1419 \fi
1420 \else
1421 \l@dttempcntb=\line@margin
1422 \ifnum\l@dttempcntb>\@ne
1423 \advance\l@dttempcntb \page@num
1424 \fi
1425 \ifodd\l@dttempcntb
1426 \gdef\l@drd@ta{\rlap{\rightpstartnumL}}%
1427 \else
1428 \gdef\l@dld@ta{\llap{\leftpstartnumL}}%
1429 \fi
1430 \fi
1431 \fi
1432 }
1433 \newcommand*{\affixpstart@numR}{%
1434 \ifsidepstartnum
1435 \if@twocolumn
1436 \if@firstcolumn
1437 \gdef\l@dld@ta{\llap{\leftpstartnumR}}%
1438 \else
1439 \gdef\l@drd@ta{\rlap{\rightpstartnumR}}%
1440 \fi
1441 \else

```

```

1442 \l@l@tempcntb=\line@marginR
1443 \ifnum\l@l@tempcntb>\@ne
1444 \advance\l@l@tempcntb \page@numR
1445 \fi
1446 \ifodd\l@l@tempcntb
1447 \gdef\l@d@rd@ta{\rlap{{\rightstartnumR}}}%
1448 \else
1449 \gdef\l@d@d@ta{\llap{{\leftstartnumR}}}%
1450 \fi
1451 \fi
1452 \fi
1453 }
1454
1455 \newcommand*{\leftstartnumL}{
1456 \ifpstartnum
1457 \thepstartL
1458 \kern\linenumsep\global\pstartnumfalse\fi
1459 }
1460 \newcommand*{\rightstartnumL}{
1461 \ifpstartnum\kern\linenumsep
1462 \thepstartL
1463 \global\pstartnumfalse\fi
1464 }
1465 \newif\ifpstartnumR
1466 \pstartnumRtrue
1467 \newcommand*{\leftstartnumR}{
1468 \ifpstartnumR
1469 \thepstartR
1470 \kern\linenumsep\global\pstartnumRfalse\fi
1471 }
1472 \newcommand*{\rightstartnumR}{
1473 \ifpstartnumR\kern\linenumsep
1474 \thepstartR
1475 \global\pstartnumRfalse\fi
1476 }
1477 %

```

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.

```

1478 \list@create{\inserts@listR}
1479 %

```

`\add@insertsR` The right text version.

```

\add@inserts@nextR
1480 \newcommand*{\add@insertsR}{%
1481 \global\let\add@inserts@nextR=\relax

```

```

1482 \ifx\inserts@listR\empty \else
1483 \ifx\next@insertR\empty
1484 \ifx\insertlines@listR\empty
1485 \global\noteschanged@true
1486 \gdef\next@insertR{100000}%
1487 \else
1488 \gl@p\insertlines@listR\to\next@insertR
1489 \fi
1490 \fi
1491 \ifnum\next@insertR=\absline@numR
1492 \gl@p\inserts@listR\to\@insertR
1493 \@insertR
1494 \global\let\@insertR=\undefined
1495 \global\let\next@insertR=\empty
1496 \global\let\add@inserts@nextR=\add@insertsR
1497 \fi
1498 \fi
1499 \add@inserts@nextR}
1500
1501 %

```

VIII.7 Penalties

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

In the code below, which is a virtual copy of the original `\add@penalties`, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we are working on at the moment. The count `\@l@tempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast`. Finally, the penalty is checked to see that it does not go below -10000 .

```

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

```

```

\ifnum\@l@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.

```

1502 \newcommand*\add@penaltiesL}{%
1503 \newcommand*\add@penaltiesR}{%
1504
1505 %

```

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.

```

1506 \newcommand*\flush@notesR}{%
1507   \@xloop
1508   \ifx\inserts@listR\empty \else
1509     \glp\inserts@listR\to\@insertR
1510     \@insertR
1511     \global\let\@insertR=\undefined
1512   \repeat}
1513
1514 %

```

IX Footnotes

IX.1 Footnotes output specific to \Pages

\print@Xnotes@forpages The `\Xonlyside` and `\onlysideX` hooks for `\Pages` allow notes to be printed either in left or right pages only. The implementation of such features is delegated to **\correct@Xfootins@box** `\print@notesX@forpages`, which replaces `\print@Xnotes` inside `\Pages`. Here is how we proceed⁶:

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

- If notes are to be printed in both sides, we just proceed the usual way: print the foot starts for the series, then the foot group.
- If notes are to be printed in the left side, we do these prints only for even pages ; if notes are to be printed in the right side, we do these prints only for odd pages.
- However, that is not enough. Because the problem does not only consists in printing notes in any particular page. It is also not to put aside room for notes in the pages where we do not want to print them. To take an example: if some note in the left side is too long by 160pt to be printed in full in the left page, we do not want to put aside 160pt a space for it in the following right page.
- To solve this problem, we change the magnification factor associated with notes before going to the next page. If we start a page where no notes are supposed to be printed, the magnification counter is set to 0. We also set the note skip to 0pt. Before starting a new page where these notes are supposed to be printed, we reset these counter and skip to their default values. (About these counter and skip, read *The TeXbook* p. 122-125).
- There still remains a last problem. This problem is quite complex to understand, so an example will speak for itself. Suppose we allow 10 lines of notes by page. Suppose a long note, be it 25 lines, which needs three pages to be printed. Suppose it must be printed only on left pages, namely odd pages.

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

To fix this double issue:

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

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

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

```
1515 \newcommand\print@Xnotes@forpages[1]{%
1516 %
```

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

```
1517 \ifcseempty{Xonlyside@#1}{%
1518 \csuse{#1footstart}{#1}%
1519 \csuse{#1footgroup}{#1}%
1520 }%
1521 %
```

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

```
1522 {%
1523 \ifboolexpr{%
1524 ((test {\ifcsstring{Xonlyside@#1}{L}} and not test{\ifnumodd{\c@page
1525 }})%
1526 or%
1527 (test {\ifcsstring{Xonlyside@#1}{R}} and test{\ifnumodd{\c@page}}))%
1528 }%
1529 %
```

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

```
1529 {%
1530 \correct@Xfootins@box{#1}%
1531 \csuse{#1footstart}{#1}%
1532 \csuse{#1footgroup}{#1}%
1533 %
```

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

```
1534 \global\count\csuse{#1footins}=0%
1535 \global\skip\csuse{#1footins}=0pt%
1536 %
```

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

```
1537 \csuse{Xnotefontsize@#1}%
1538 \global\advance\dimen\csuse{#1footins} by -\baselineskip%
1539 %
```

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

```
1540 \global\boolfalse{keepforXside@#1}%
1541 }%
1542 %
```

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.

```

1543 {%
1544   \global\booltrue{keepforXside@#1}%
1545 %

```

Then restore expected rooms for notes on the next page.

```

1546   \global\count\csuse{#1footins}=\csuse{default@#1footins}%
1547   \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@#1}%
1548 %

```

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

```

1549   \bgroup%
1550   \csuse{Xnotefontsize@#1}%
1551   \global\advance\dimen\csuse{#1footins} by \baselineskip%
1552   \egroup%
1553 %

```

```

1554 % End of \protect\cs{print@Xnotes@forpages}.
1555 }%
1556 }%
1557 }%
1558 %

```

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

```

1559 \newcommand{\correct@Xfootins@box}[1]{%
1560 %

```

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.

```

1561   \ifbool{keepforXside@#1}{%
1562 %

```

Some setting needed to do the right splitting.

```

1563   \csuse{Xnotefontsize@#1}%
1564   \splittopskip=0pt%
1565 %

```

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

```

1566   \global\setbox\csuse{#1footins}=\vbox{%
1567     \vsplit\csuse{#1footins} to \dimexpr\ht\csuse{#1footins}-1pt\relax%
1568     \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1569     \unvbox\csuse{#1footins}%
1570   }%
1571 %

```

End of the macro.

```

1572 }{}%
1573 }%
1574 %

```

And now, the same for familiar footnotes.

```

1575 \newcommand\print@notesX@forpages[1]{%
1576   \ifcseempty{onlysideX@#1}{%
1577     \csuse{footstart#1}{#1}%
1578     \csuse{footgroup#1}{#1}%
1579   }%
1580   {%
1581     \ifboolexpr{%
1582       ((test {\ifcsstring{onlysideX@#1}{L}} and not test{\ifnumodd{\c@page
1583       }%
1584       (test {\ifcsstring{onlysideX@#1}{R}} and test{\ifnumodd{\c@page}}))%
1585     }%
1586     {%
1587       \correct@footinsX@box{#1}%
1588       \csuse{footstart#1}{#1}%
1589       \csuse{footgroup#1}{#1}%
1590       \global\count\csuse{footins#1}=0%
1591       \global\skip\csuse{footins#1}=0pt%
1592       \csuse{notefontsizeX@#1}%
1593       \global\advance\dimen\csuse{footins#1} by -\baselineskip%
1594       \global\boolfalse{keepforsideX@#1}%
1595     }%
1596     {%
1597       \global\booltrue{keepforsideX@#1}%
1598       \global\count\csuse{footins#1}=\csuse{default@footins#1}%
1599       \global\skip\csuse{footins#1}=\csuse{beforenotesX@#1}%
1600       \bgroup%
1601         \csuse{notefontsizeX@#1}%
1602         \global\advance\dimen\csuse{footins#1} by \baselineskip%
1603       \egroup%
1604     }%
1605   }%
1606 }%
1607 \newcommand{\correct@footinsX@box}[1]{%
1608   \ifbool{keepforsideX@#1}{%
1609     \csuse{notefontsizeX@#1}%
1610     \splittopskip=0pt%
1611     \global\setbox\csuse{footins#1}=\vbox{%
1612       \vsplit\csuse{footins#1} to \dimexpr\ht\csuse{footins#1}-1pt\relax%
1613       \vskip \dimexpr-0.5\baselineskip-0.5\lineskip-0.5pt\relax%
1614       \unvbox\csuse{footins#1}%
1615     }%
1616   }{}%
1617 }%
1618 %

```


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.

```
1619 \list@create{\labelref@listR}
1620
1621 %
```

`\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`.

```
1622 \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1623   \expandafter\ifx\csname the@label#5\endcsname \relax\else
1624     \led@warn@DuplicateLabel{#5}%
1625   \fi
1626   \expandafter\gdef\csname the@label#5\endcsname{#1|#2|#3|#4|\@Rlineflag}%
1627   \ignorespaces}
1628 \AtBeginDocument{%
1629   \def\l@dmake@labelsR#1|#2|#3|#4|#5{%
1630   }
1631
1632 %
```

`\@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*
1633 \WithSuffix\newcommand\sidenotemargin*[1]{%
1634   \l@dgetsidenote@margin{#1}
1635   \global\sidenote@marginR=\@l@dttempcntb
1636   \global\sidenote@margin=\@l@dttempcntb
1637 }
1638 \newcount\sidenote@marginR
1639 \global\sidenote@margin=\@ne
1640
1641 %
```

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

```

1642 \newcommand*{\affixside@noteR}{%
1643   \def\sidenotecontent@{%
1644     \numgdef{\itemcount@}{0}%
1645     \def\do##1{%
1646       \ifnumequal{\itemcount@}{0}%
1647         {%
1648           \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
1649         {\appto\sidenotecontent@{\sidenotesep ##1}%
1650         }%
1651         \numgdef{\itemcount@}{\itemcount@+1}%
1652       }%
1653       \dolistloop{\l@dcstotetext}%
1654       \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
1655       \gdef\@templ@d{%
1656         \gdef\@templ@n{\l@dcstotetext\l@dcstotetext@l\l@dcstotetext@r}%
1657         \ifx\@templ@d\@templ@n \else%
1658           \if@twocolumn%
1659             \if@firstcolumn%
1660               \setl@dlp@rbox{##1}{\sidenotecontent@}%
1661             \else%
1662               \setl@drp@rbox{\sidenotecontent@}%
1663             \fi%
1664           \else%
1665             \l@dttempcntb=\sidenote@marginR%
1666             \ifnum\l@dttempcntb>\@ne%
1667               \advance\l@dttempcntb by\page@numR%
1668             \fi%
1669             \ifodd\l@dttempcntb%
1670               \setl@drp@rbox{\sidenotecontent@}%
1671             \gdef\sidenotecontent@{%
1672               \numdef{\itemcount@}{0}%
1673               \dolistloop{\l@dcstotetext@l}%
1674               \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
1675               \setl@dlp@rbox{\sidenotecontent@}%
1676             \else%
1677               \setl@dlp@rbox{\sidenotecontent@}%
1678             \gdef\sidenotecontent@{%
1679               \numdef{\itemcount@}{0}%
1680               \dolistloop{\l@dcstotetext@r}%
1681               \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
1682               \setl@drp@rbox{\sidenotecontent@}%
1683             \fi%
1684           \fi%
1685         \fi%
1686       }
1687     }%
1688   }
```

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.

```

\inserthangingsymbolL89 \newif\ifinserthangingsymbolR
\inserthangingsymbolR90 \newcommand{\inserthangingsymbolL}{%
1691   \ifinserthangingsymbol%
1692     \ifinstanzaL%
1693       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1694         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1695       \@hangingsymbol%
1696     \fi%
1697   \fi%
1698 }%
1699 \newcommand{\inserthangingsymbolR}{%
1700   \ifinserthangingsymbolR%
1701     \ifinstanzaR%
1702       \hskip \@ifundefined{sza@00}{0}{\expandafter%
1703         \noexpand\csname sza@00\endcsname}\stanzaindentbase%
1704       \@hangingsymbol%
1705     \fi%
1706   \fi%
1707 }%
1708 %

```

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.

```

1709 \chardef\next=\catcode`\&
1710 \catcode`\&=\active
1711
1712 %

```

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

```

1713 \newenvironment{astanza}[1][\catcode`\&=\active]{%
1714   \catcode`\&=\active
1715   \global\stanza@count\@ne\stanza@modulo\@ne

```

```

1716 \ifnum\usernamecount{sza@0@}=\z@
1717 \let\stanza@hang\relax
1718 \let\endlock\relax
1719 \else
1720 \rightskip\z@ plus 1fil\relax
1721 \fi
1722 \ifnum\usernamecount{szp@0@}=\z@
1723 \let\sza@penalty\relax
1724 \fi
1725 \def&{%
1726 \endlock\mbox{}%
1727 \sza@penalty
1728 \global\advance\stanza@count\@ne
1729 \@astanza@line}%
1730 \def\&{\@stopastanza}%
1731 \pstart[#1]%
1732 \@astanza@line
1733 \let\par\relax\ignorespaces%No paragraph in verses
1734 }{}
1735
1736 %

```

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

```

1737 \newcommand{\@stopastanza}[1][1,usedefault]{%
1738 \endlock\mbox{}%
1739 \pend[#1]%
1740 }%
1741 %

```

\@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.

```

1742 \newcommand*{\@astanza@line}{%
1743 \ifnum\value{stanzaindentrepetition}=0
1744 \parindent=\csname sza@\number\stanza@count
1745 @\endcsname\stanzaindentbase
1746 \else
1747 \parindent=\csname sza@\number\stanza@modulo
1748 @\endcsname\stanzaindentbase
1749 \managestanza@modulo
1750 \fi
1751 \endgraf
1752 \stanza@hang%
1753 \ignorespaces}
1754
1755 %

```

Lastly reset the modified category codes.

```

1756 \catcode`\&=\next
1757
1758 %

```

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

```

\thestanzaR
1759 \newcounter{stanzaL}
1760 \newcounter{stanzaR}
1761 \renewcommand{\thestanzaL}{%
1762   \textbf{\arabic{stanzaL}}%
1763 }
1764 \renewcommand{\thestanzaR}{%
1765   \textbf{\arabic{stanzaR}}%
1766 }
1767 %
1768 %

```

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
1769 \providecommand*\newnamebox}[1]{%
1770   \expandafter\newbox\csname #1\endcsname}
1771 \providecommand*\setnamebox}[1]{%
1772   \expandafter\setbox\csname #1\endcsname}
1773 \providecommand*\unhnamebox}[1]{%
1774   \expandafter\unhbox\csname #1\endcsname}
1775 \providecommand*\unvnamebox}[1]{%
1776   \expandafter\unvbox\csname #1\endcsname}
1777 \providecommand*\namebox}[1]{%
1778   \csname #1\endcsname}
1779
1780 %

```

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

```

\usenamecount
1781 \providecommand*\newnamecount}[1]{%
1782   \expandafter\newcount\csname #1\endcsname}
1783 \providecommand*\usenamecount}[1]{%
1784   \csname #1\endcsname}
1785
1786 %

```

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@dusedbabeltrue 1787 \newif\ifl@dusedbabel
1788 %
```

```
\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 \.
```

```
1789 \patchcmd{\bbl@set@language}%
1790 {\select@language{\language}}%
1791 {\edef\language{#1}\select@language{\language}}%
1792 {}%
1793 {}%
1794
1795 %
```

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
1796 \newcommand*{\l@duselanguage}[1]{%
1797 \gdef\theledlanguageL{
1798 \gdef\theledlanguageR{
1799
1800 %
```

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

```

1801 \AtBeginDocument{%
1802   \ifundefined{xpg@main@language}{%
1803     \ifundefined{bbl@main@language}{%
1804 %

```

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

```

1805   \l@dusedbabelfalse
1806 }{%
1807 %

```

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.

```

1808   \l@dusedbabeltrue
1809   \let\l@doldselectlanguage\selectlanguage
1810   \let\l@doldbbl@set@language\bbl@set@language
1811   \renewcommand{\selectlanguage}[1]{%
1812     \l@doldselectlanguage{#1}%
1813     \ifledRcol \gdef\theledlanguageR{#1}%
1814     \else      \gdef\theledlanguageL{#1}%
1815     \fi}
1816 %

```

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

```

1817   \renewcommand*{\l@duselanguage}[1]{%
1818     \expandafter\l@doldselectlanguage\expandafter{#1}}
1819 %

```

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

```

1820   \gdef\theledlanguageL{\bbl@main@language}%
1821   \gdef\theledlanguageR{\bbl@main@language}%
1822 }%
1823 }
1824 %

```

If use polyglossia

```

1825 { \let\old@otherlanguage\otherlanguage%
1826   \renewcommand{\otherlanguage}[2] []{%
1827     \selectlanguage{#1}{#2}%
1828     \ifledRcol \gdef\theledlanguageR{#2}%
1829     \else      \gdef\theledlanguageL{#2}%
1830     \fi}%
1831   \let\l@duselanguage\select@language%
1832   \gdef\theledlanguageL{\xpg@main@language}%
1833   \gdef\theledlanguageR{\xpg@main@language}%
1834 %

```

That is it.

```
1835 }}
1836 %
```

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`

```
1837 \newcount\l@dc@maxchunks
1838 \newcommand{\maxchunks}[1]{\l@dc@maxchunks=#1}
1839 \maxchunks{5120}
1840
1841 %
```

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

```
1842 \newcount\l@dnumpstartsR
1843
1844 %
```

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

`\l@pscR`

```
1845 \newcount\l@dpscl
1846 \newcount\l@dpscR
1847
1848 %
```

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

```
1849 \newcommand*\l@dsetuprawboxes{%
1850 \l@dttempcntb=\l@dc@maxchunks
1851 \loop\ifnum\l@dttempcntb>\z@
1852 \newnamebox{\l@dLcolrawbox\the\l@dttempcntb}
1853 \newnamebox{\l@dRcolrawbox\the\l@dttempcntb}
1854 \advance\l@dttempcntb \m@ne
1855 \repeat}
1856
1857 %
```

`\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.


```

1858 \newcommand*{\l@dssetupmaxlinecounts}{%
1859   \l@ds@tempcntb=\l@ds@maxchunks
1860   \loop\ifnum\l@ds@tempcntb>\z@
1861     \newnamecount{\l@ds@maxlinesinpar\the\l@ds@tempcntb}
1862     \advance\l@ds@tempcntb \m@ne
1863   \repeat}
1864 \newcommand*{\l@ds@zeromaxlinecounts}{%
1865   \begin{group}
1866     \l@ds@tempcntb=\l@ds@maxchunks
1867     \loop\ifnum\l@ds@tempcntb>\z@
1868       \global\usenamecount{\l@ds@maxlinesinpar\the\l@ds@tempcntb}=\z@
1869       \advance\l@ds@tempcntb \m@ne
1870     \repeat
1871   \end{group}}
1872
1873 %

```

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

```

1874 \AtBeginDocument{%
1875   \l@ds@setuprawboxes
1876   \l@ds@setupmaxlinecounts
1877   \l@ds@zeromaxlinecounts
1878   \l@ds@dumpstartsL=\z@
1879   \l@ds@dumpstartsR=\z@
1880   \l@ds@psscL=\z@
1881   \l@ds@psscR=\z@}
1882
1883 %

```

XVII Checking text to be processed

```

\if@pstarts \check@pstarts returns \@pstartstrue if there are any unprocessed chunks.
\@pstartstrue
\@pstartfalse
\check@pstarts
1884 \newif\if@pstarts
1885 \newcommand*{\check@pstarts}{%
1886   \@pstartfalse
1887   \ifnum\l@ds@dumpstartsL>\l@ds@psscL
1888     \@pstartstrue
1889   \else
1890     \ifnum\l@ds@dumpstartsR>\l@ds@psscR
1891       \@pstartstrue
1892     \fi
1893   \fi
1894 }
1895
1896 %

```

`\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`

```

1897 \newif\ifaraw@text
1898 \newcommand*\checkraw@text{%
1899   \araw@textfalse
1900   \ifvbox\namebox{1@dLcolrawbox\the\l@dpscl}
1901     \araw@texttrue
1902   \else
1903     \ifvbox\namebox{1@dRcolrawbox\the\l@dpscl}
1904       \araw@texttrue
1905     \fi
1906   \fi
1907 }
1908
1909 %

```

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

`\@writelinesinparR`

```

1910 \newcommand*\@writelinesinparL{%
1911   \edef\next{%
1912     \write\linenum@out{\string\@pend[\the\@donereallinesL]}}%
1913   \next
1914   \global\@donereallinesL \z@
1915 \newcommand*\@writelinesinparR{%
1916   \edef\next{%
1917     \write\linenum@outR{\string\@pendR[\the\@donereallinesR]}}%
1918   \next
1919   \global\@donereallinesR \z@
1920
1921 %

```

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

`\@writepageofparR`

```

1922 \newcommand*\@writepageofparL[0]{%
1923   \ifnum\@donereallinesL=\z@%
1924     \edef\next{%
1925       \write\linenum@out{\string\@pstart{\the\l@dpscl}{\the\c@page}{\the\
1926         numpagelinesL}}%
1927     }%
1928     \next%
1929   \fi%
1930 }%
1931 \newcommand*\@writepageofparR[0]{%
1932   \ifnum\@donereallinesR=\z@%
1933     \edef\next{%
1934       \write\linenum@outR{\string\@pstartR{\the\l@dpscl}{\the\c@page}{\the\
1935         numpagelinesR}}%

```

```

1934 }%
1935 \next%
1936 \fi%
1937 }%
1938 %

```

XVIII Parallel columns

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

```

\@eledsectionR
1939 \newsavebox{\@eledsectionL}%
1940 \newsavebox{\@eledsectionR}%
1941 %

```

`\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.

```

1942 \newcommand*{\Columns}{%
1943   \ifl@dpairing%
1944     \led@err@Columns@InsideEnv%
1945   \fi%
1946   \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname%
1947     \led@err@Columns@WithoutEnv%
1948   \else%
1949     \global\l@dprintingcolumnstrue%
1950     \eledsection@correcting@skip=-\baselineskip% Correction for sections'
titles
1951     \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else
1952       \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
1953     \fi
1954   %

```

Start a group and zero counters, etc.

```

1955 \begingroup
1956   \l@dzeropenalties
1957   \endgraf\global\l@num@lines=\prevgraf
1958   \global\l@num@linesR=\prevgraf
1959   \global\l@par@line=\z@
1960   \global\l@par@lineR=\z@
1961   \global\l@dpscL=\z@
1962   \global\l@dpscR=\z@
1963   \get@familiarfootnote@number%
1964 %

```

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

```

1965 \check@pstarts
1966 \loop\if@pstarts

```

```

1967 \global\pstartnumtrue
1968 \global\pstartnumRtrue
1969 %

```

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.

```

1970 \global\advance\l@dpscL \@ne
1971 \global\advance\l@dpscR \@ne
1972 \restore@pstartL@pc%
1973 \restore@pstartR@pc%
1974 %

```

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

```

1975 \Columns@print@before@pstart%
1976 %

```

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

```

1977 \checkraw@text
1978 { \loop\ifaraw@text
1979 %

```

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

```

1980 \l@duselanguage{\theledlanguageL}%
1981 \do@lineL
1982 \xifinlist{\the\l@dpscL}{\eled@sections@@}
1983 {%
1984 \ifdefstring{\@eledsectmark}{L}%
1985 {\csuse{eled@sectmark@the\l@dpscL}%
1986 }}%
1987 \global\csundef{eled@sectmark@the\l@dpscL}%
1988 \savebox{\@eledsectionL}{\parbox[t]{}[t]{\Lcolwidth}{\vbox
1989 {} \print@eledsectionL}}}%\vbox{}-> prevent alignment troubles with RTL
1990 language
1991 }%
1992 {}%
1993 \l@duselanguage{\theledlanguageR}%
1994 \do@lineR
1995 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}
1996 {%
1997 \ifdefstring{\@eledsectmark}{R}%
1998 {\csuse{eled@sectmark@the\l@dpscR}%
1999 }}%
2000 \global\csundef{eled@sectmark@the\l@dpscR}%
2001 \savebox{\@eledsectionR}{\parbox[t]{}[t]{\Rcolwidth}{\vbox
2002 {} \print@eledsectionR}}}%\vbox{}-> prevent alignment troubles with RTL
2003 language
2004 {}%

```

```

2001 \hb@xt@ \hsize{%
2002 \ifdefstring{\columns@position}{L}{\hfill }%
2003 \unhbox\l@leftbox%
2004 \ifhbox\@eledsectionL%
2005 \usebox{\@eledsectionL}%
2006 \fi%
2007 \print@columnseparator%
2008 \unhbox\l@drightbox%
2009 \ifhbox\@eledsectionR%
2010 \usebox{\@eledsectionR}%
2011 \fi%
2012 \ifdefstring{\columns@position}{R}{\hfill}%
2013 }%
2014 \checkraw@text
2015 \checkverseL
2016 \checkverseR
2017 \checkpb@columns
2018 \repeat}
2019 %

```

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`.

```

2020 \@writelinesinparL
2021 \@writelinesinparR
2022 \check@pstarts
2023 \ifbypstart@%
2024 \write\linenum@out{\string\@set[1]}
2025 \resetprevline@
2026 \fi
2027 \ifbypstart@R
2028 \write\linenum@outR{\string\@set[1]}
2029 \resetprevline@
2030 \fi
2031 \Columns@print@after@pend%
2032 \repeat
2033 %

```

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.

```

2034 \flush@notes
2035 \flush@notesR
2036 \endgroup
2037 %

```

```

2038 \global\l@dpscl=\z@
2039 \global\l@dpscr=\z@
2040 \global\l@dnpstartsL=\z@
2041 \global\l@dnpstartsR=\z@

```

```

2042 \if@ledgroup\else%
2043   \global\l@printingcolumnsfalse%
2044 \fi
2045 \ignorespaces
2046   \global\instanzaLfalse%
2047   \global\instanzaRfalse%
2048 \fi}
2049
2050 %

```

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

```

2051 \def\print@columnseparator{%
2052   \ifdim\beforecolumnseparator<0pt%
2053     \hfill%
2054   \else%
2055     \hspace{\beforecolumnseparator}%
2056   \fi%
2057   \columnseparator%
2058   \ifdim\aftercolumnseparator<0pt%
2059     \hfill%
2060   \else%
2061     \hspace{\aftercolumnseparator}%
2062   \fi%
2063 }%
2064 %

```

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

```

2065
2066 \newcommand{\checkpb@columns}{%
2067   \newif\if@pb
2068   \newif\if@nopb
2069   \IfStrEq{\led@pb@setting}{before}{
2070     \numdef{\next@absline}{\the\absline@num+1}%
2071     \numdef{\next@abslineR}{\the\absline@numR+1}%
2072     \xifinlistcs{\next@absline}{l@prev@pb}{\@pbtrue}{}%
2073     \xifinlistcs{\next@abslineR}{l@prev@pbR}{\@pbtrue}{%
2074       \xifinlistcs{\next@absline}{l@prev@nopb}{\@nopbtrue}{}%
2075       \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\@nopbtrue}{%
2076         }{}
2077     \IfStrEq{\led@pb@setting}{after}{
2078       \xifinlistcs{\the\absline@num}{l@prev@pb}{\@pbtrue}{}%
2079       \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\@pbtrue}{%
2080       \xifinlistcs{\the\absline@num}{l@prev@nopb}{\@nopbtrue}{}%
2081       \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\@nopbtrue}{%
2082         }{}

```

```

2083 \if@nopb\nopagebreak[4]\enlargethispage{\baselineskip}\fi
2084 \if@pb\pagebreak[4]\fi
2085 }
2086 %

```

\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).

```

2087 \newcommand*{\columnseparator}{%
2088   \smash{\rule[-0.2\baselineskip]{\columnrulewidth}{1.05\baselineskip}}
2089   \newdimen\columnrulewidth
2090   \columnrulewidth=\z@
2091 }
2092 %

```

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

```

\columns@position
2093 \newcommand*{\columnspan}[1]{%
2094   \xdef\columns@position{#1}%
2095 }%
2096 \xdef\columns@position{R}%
2097 %

```

\beforecolumnseparator **\aftercolumnseparator** **\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**.

```

2098 \newlength{\beforecolumnseparator}%
2099 \setlength{\beforecolumnseparator}{-2pt}%
2100
2101 \newlength{\aftercolumnseparator}%
2102 \setlength{\aftercolumnseparator}{-2pt}%
2103
2104 %

```

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

```

\setwidthliketwocolumns@L
2105 \newcommand{\setwidthliketwocolumns@L}{%
2106   % Temporary dimension, initially equal to the standard hsize, i.e. text
2107   width
2108   \begin{macrocode}
2109   \newdimen\temp%
2110   \temp=\hsize%
2111 }

```

Hsize : Left + Right width

```
2111 \hsize=\Lcolwidth%
2112 \advance\hsize\Rcolwidth%
2113 %
```

Now, calculating the remaining space

```
2114 \advance\temp-\hsize%
2115 %
```

And multiply the hsize by 2/3 of this space

```
2116 \multiply\temp by 2%
2117 \divide\temp by 3%
2118 \advance\hsize\temp%
2119 }%
2120
2121 \newcommand{\setpositionliketwocolumns@L}{%
2122 \renewcommand{\ledrlfill}{\hfill}%
2123 }%
2124
2125 \newcommand{\setnotespositionliketwocolumns@L}{%
2126 }%
2127
2128
2129 %
```

```
2130 \newcommand{\setwidthliketwocolumns@C}{%
2131 % Temporary dimension, initially equal to the standard hsize, i.e. text
width
2132 %
```

```
2133 \newdimen\temp%
2134 \temp=\hsize%
2135 % Hsize : Left + Right width
2136 %
```

```
2137 \hsize=\Lcolwidth%
2138 \advance\hsize\Rcolwidth%
2139 % Now, calculating the remaining space
2140 %
```

```
2141 \advance\temp-\hsize%
2142 %
```

And multiply the hsize by 1/2 of this space

```
2143 \divide\temp by 2%
2144 \advance\hsize\temp%
2145 }%
2146
2147 \newcommand{\setpositionliketwocolumns@C}{%
```



```

2148 \doinsidelinehook{\hfill}%
2149 \renewcommand{\ledrlfill}{\hfill}%
2150 }%
2151
2152 \newcommand{\setnotespositionliketwocolumns@C}{%
2153 \newdimen\temp%
2154 \newdimen\tempa%
2155 \temp=\hsize%
2156 \tempa=\Lcolwidth%
2157 \advance\tempa\Rcolwidth%
2158 \advance\temp-\tempa%
2159 \divide\temp by 2%
2160 \leftskip=\temp%
2161 \rightskip=-\temp%
2162 }%
2163
2164 \newcommand{\setwidthliketwocolumns@R}{%
2165 %

```

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

```

2166 \newdimen\temp%
2167 \temp=\hsize%
2168 %

```

Hsize : Left + Right width

```

2169 \hsize=\Lcolwidth%
2170 \advance\hsize\Rcolwidth%
2171 %

```

Now, calculating the remaining space

```

2172 \advance\temp-\hsize%
2173 %

```

And multiply the hsize by 2/3 of this space

```

2174 \multiply\temp by 2%
2175 \divide\temp by 3%
2176 \advance\hsize\temp%
2177 }%
2178
2179 \newcommand{\setpositionliketwocolumns@R}{%
2180 \doinsidelinehook{\hfill}%
2181 }%
2182
2183 \newcommand{\setnotespositionliketwocolumns@R}{%
2184 \newdimen\temp%
2185 \newdimen\tempa%
2186 \temp=\hsize%
2187 \tempa=\Lcolwidth%
2188 \advance\tempa\Rcolwidth%

```

```

2189 \advance\temp-\tempa%
2190 \divide\temp by 2%
2191 \leftskip=\temp%
2192 \rightskip=-\temp%
2193 }%
2194
2195 %

```

`\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.

```

2196 \newcommand{\Columns@print@before@pstart}{%
2197   \ifboolexpr{%
2198     test{\ifcsstring{before@pstartL@the\l@dpscl}{\at@every@pstart}}%
2199     and test {\ifcsstring{before@pstartR@the\l@dpscl}{\at@every@pstart}}%
2200     and test {\ifdefempty{\at@every@pstart}}}%
2201     {}%
2202     {%
2203       \hb@xt@ \hsize{%
2204         \ifdefstring{\columns@position}{L}{\hfill }%
2205         \par\parbox[t]{}[t]{\Lcolwidth}{%
2206           \csuse{before@pstartL@the\l@dpscl}%
2207         }%
2208         \print@columnseparator%
2209         \parbox[t]{}[t]{\Rcolwidth}{%
2210           \set@sectcountR%
2211           \csuse{before@pstartR@the\l@dpscl}%
2212         }%
2213         \ifdefstring{\columns@position}{R}{\hfill}%
2214       }%
2215     }%
2216   \global\csundef{before@pstartL@the\l@dpscl}%
2217   \global\csundef{before@pstartR@the\l@dpscl}%
2218 }%
2219 \newcommand{\Columns@print@after@pend}{%
2220   \ifboolexpr{%
2221     test{\ifcsstring{after@pendL@the\l@dpscl}{\at@every@pend}}%
2222     and test {\ifcsstring{after@pendR@the\l@dpscl}{\at@every@pend}}%
2223     and test {\ifdefempty{\at@every@pend}}}%
2224     {}%
2225     {%
2226       \hb@xt@ \hsize{%
2227         \ifdefstring{\columns@position}{L}{\hfill }%
2228         \parbox[t]{}[t]{\Lcolwidth}{%
2229           \csuse{after@pendL@the\l@dpscl}%
2230         }%
2231         \print@columnseparator%
2232         \parbox[t]{}[t]{\Rcolwidth}{%
2233           \set@sectcountR%

```

```

2234         \csuse{after@pendR@the\l@dpscR}%
2235     }%
2236     \ifdefstring{\columns@position}{R}{\hfill}%
2237 }%
2238 }%
2239 \global\csundef{after@pendL@the\l@dpscL}%
2240 \global\csundef{after@pendR@the\l@dpscR}%
2241 }%
2242 %

```

XIX Parallel pages

This is considerably more complicated than parallel columns.

XIX.1 Specific counters

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

```

2243 \newcount\numpagelinesL
2244 \newcount\numpagelinesR
2245 \newcount\l@dminpagelines
2246
2247 %

```

XIX.2 Main macro

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

```

2248 \newcommand*{\Pages}[1][1,usedefault]{%
2249     \ifl@dpairing%
2250         \led@err@Pages@InsideEnv%
2251     \fi%
2252     \expandafter\ifvoid\csname l@dRcolrawbox1\endcsname%
2253         \led@err@Pages@WithoutEnv%
2254     \else%
2255         \ifstrequal{#1}{mainmatter}{\Pages@mainmattertrue}{\Pages@mainmatterfalse}%
2256     }%
2257     \eledsection@correcting@skip=-2\baselineskip% line correcting for section
2258     titles.
2259     \parledgroup@notespacing@set@correction%
2260     \typeout{}%
2261     \typeout{***** PAGES *****}%
2262     \ifnum\l@dnumpstartsL=\l@dnumpstartsR\else%
2263         \led@err@BadLeftRightPstarts{\the\l@dnumpstartsL}{\the\l@dnumpstartsR}%
2264     \fi%

```

2263 %

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

```
2264 \cleartol@devenpage%
2265 \global\l@dprintingpagetrue%
2266 \begingroup%
2267 %
```

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

```
2268 \setlength{\Lcolwidth}{\textwidth}%
2269 \setlength{\Rcolwidth}{\textwidth}%
2270 %
```

```
2271 \l@dzeropenalties%
2272 \endgraf\global\num@lines=\prevgraf%
2273 \global\num@linesR=\prevgraf%
2274 \global\par@line=\z@%
2275 \global\par@lineR=\z@%
2276 \global\l@dpscL=\z@%
2277 \global\l@dpscR=\z@%
2278 \writtenlinesLfalse%
2279 \writtenlinesRfalse%
2280 \get@familiarfootnote@number%
2281 %
```

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

```
2282 \let\print@Xnotes\print@Xnotes@forpages%
2283 \let\print@notesX\print@notesX@forpages%
2284 %
```

Check if there are chunks to be processed.

```
2285 \check@pstarts%
2286 \loop\if@pstarts%
2287 %
```

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

```
2288 \global\advance\l@dpscL \@ne%
2289 \global\advance\l@dpscR \@ne%
2290 %
```

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

```
2291 \getlinesfromparlistL%
2292 \getlinesfromparlistR%
2293 \l@dcalc@maxoftwo{\@cs@linesinparL}{\@cs@linesinparR}%
2294 {\usernamecount{l@dmaxlinesinpar\the\l@dpscL}}%
```

```

2295     \check@pstarts%
2296     \repeat%
2297 %

```

Zero the counts again, ready for the next bit.

```

2298     \global\l@dpscL=\z@%
2299     \global\l@dpscR=\z@%
2300 %

```

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

```

2301     \getlinesfrompagelistL%
2302     \getlinesfrompagelistR%
2303     \l@dcalc@minoftwo{\cs@linesonpageL}{\cs@linesonpageR}%
2304         {\l@dminpagelines}%
2305 %

```

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.

```

2306     \check@pstarts%
2307     \if@pstarts%
2308 %

```

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

```

2309     \global\advance\l@dpscL \@ne%
2310     \global\advance\l@dpscR \@ne%
2311     \restore@pstartL@pc%
2312     \restore@pstartR@pc%
2313 %

```

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

```

2314     \global\@donereallinesL=\z@%
2315     \global\@donetotallinesL=\z@%
2316     \global\@donereallinesR=\z@%
2317     \global\@donetotallinesR=\z@%
2318 %

```

Start a loop over the boxes (chunks).

```

2319     \checkraw@text%
2320 %
2321 %     \begingroup
2322 {     \loop\ifaraw@text%
2323 %

```

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

```

2324 \checkpageL%
2325 \l@duselanguage{\theledlanguageL}%
2326 { \loop\ifl@dsamepage%
2327 %

```

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

```

2328 \ifdefstring{\@eledsectnotoc}{L}{\ledsectnotoc}{}%
2329 \csuse{before@pstartL@the\l@dpscl}%
2330 \global\csundef{before@pstartL@the\l@dpscl}%
2331 \do@lineL%
2332 \xifinlist{the\l@dpscl}{\eled@sections@@}
2333 {\print@eledsectionL}%
2334 {}%
2335 \advance\numpagelinesL \@ne%
2336 %

```

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

```

2337 \ifshiftedpstarts%
2338 \ifdim\ht\l@dleftbox>0pt%
2339 \parledgroup@correction@notespacing{L}%
2340 \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2341 \else%
2342 \unless\ifadvancedshiftedpstarts%
2343 \dimen0=\pagetotal%
2344 \advance\dimen0 by \baselineskip%
2345 \global\pagetotal=\dimen0%
2346 \else%
2347 \ifnomaxlines%
2348 \numdef{\@tmp}{the\l@dpscl+1}%
2349 \ifcsdef{minpage@pstart@\@tmp}{%
2350 \ifnumless{the\c@page}{\csuse{
minpage@pstart@\@tmp}}%
2351 {\dimen0=\pagetotal%
2352 \advance\dimen0 by \baselineskip%
2353 \global\pagetotal=\dimen0%
2354 }%
2355 }%
2356 }%
2357 \fi%
2358 \fi%
2359 \fi%
2360 \else%

```

```

2361         \parledgroup@correction@notespacing{L}%
2362         \hb@xt@ \hsize{\ledstrutL\unhbox\l@dleftbox}%
2363     \fi%
2364 %

```

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

```

2365     \get@nextboxL%
2366     \global\l@dskipversenumberfalse%
2367     \ifprint@last@after@pendL%
2368         \csuse{after@pendL@the\l@dpscL}%
2369     \global\csundef{after@pendL@the\l@dpscL}%
2370     \fi%
2371     \checkpageL%
2372     \checkverseL%
2373     \checkpbl%
2374     \repeat%
2375 %

```

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.

```

2376     \ifl@dpagfull%
2377         \@writelinesonpageL{\the\numpagelinesL}%
2378     \else%
2379         \@writelinesonpageL{1000}%
2380     \fi%
2381 %

```

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.

```

2382     \numpagelinesL \z@%
2383     \parledgroup@correction@notespacing@init%
2384     \clearl@dleftpage }%
2385 %

```

Now do the same for the right text.

```

2386     \checkpageR%
2387     \l@duselanguage{\theledlanguageR}%
2388 {
2389     \loop\ifl@dsamepage%
2389         \set@sectcountR%
2390         \ifdefstring{\@eledsectnotoc}{R}{\ledsectnotoc}{}%
2391         \csuse{before@pstartR@the\l@dpscR}%
2392         \global\csundef{before@pstartR@the\l@dpscR}%
2393         \do@lineR%

```

```

2394 \xifinlist{\the\l@dpscR}{\eled@sectionsR@@}%
2395 {\print@eledsectionR}%
2396 {}%
2397 \advance\numpagelinesR \@ne%
2398 \ifshiftedpstarts%
2399 \ifdim\ht\l@drightbox>0pt%
2400 \parledgroup@correction@notespacing{R}%
2401 \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2402 \else%
2403 \unless\ifadvancedshiftedpstarts%
2404 \dimen0=\pagetotal%
2405 \advance\dimen0 by \baselineskip%
2406 \global\pagetotal=\dimen0%
2407 \else%
2408 \ifnomaxlines%
2409 \numdef{\@tmp}{\the\l@dpscR+1}%
2410 \ifcsdef{minpage@pstart@\@tmp}{%
2411 minpage@pstart@\@tmp}}%
2412 {\dimen0=\pagetotal%
2413 \advance\dimen0 by \baselineskip%
2414 \global\pagetotal=\dimen0%
2415 }%
2416 {}%
2417 }{}%
2418 \fi%
2419 \fi%
2420 \fi%
2421 \else%
2422 \parledgroup@correction@notespacing{R}%
2423 \hb@xt@ \hsize{\ledstrutR\unhbox\l@drightbox}%
2424 \fi%
2425 \get@nextboxR%
2426 \global\l@dskipversenumberRfalse%
2427 \ifprint@last@after@pendR%
2428 \csuse{after@pendR@\the\l@dpscR}%
2429 \global\csundef{after@pendR@\the\l@dpscR}%
2430 \fi%
2431 \checkpageR%
2432 \checkverseR%
2433 \checkpbR%
2434 \repeat%
2435 \ifl@dpagefull%
2436 \@writelinesonpageR{\the\numpagelinesR}%
2437 \else%
2438 \@writelinesonpageR{1000}%
2439 \fi%
2440 \numpagelinesR=\z@%
2441 \parledgroup@correction@notespacing@init%
2442 %

```


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

```
2443         \clearl@drighthpage}%
2444 %
```

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.

```
2445         \checkraw@text%
2446         \ifaraw@text%
2447             \getlinesfrompagelistL%
2448             \getlinesfrompagelistR%
2449             \l@dcalc@minoftwo{\@cs@linesonpageL}{\@cs@linesonpageR}%
2450                                 {\l@dminpagelines}%
2451         \fi%
2452     \repeat}%
2453 %
```

We have now output the text from all the chunks.

```
2454     \fi%
2455 %
```

Make sure that there are no inserts hanging around.

```
2456     \flush@notes%
2457     \flush@notesR%
2458     \endgroup%
2459 %
```

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

```
2460     \global\l@dpscL=\z@%
2461     \global\l@dpscR=\z@%
2462     \global\l@dnumpstartsL=\z@%
2463     \global\l@dnumpstartsR=\z@%
2464     \global\instanzaLfalse%
2465     \global\instanzaRfalse%
2466     \if@ledgroup\else%
2467         \global\l@dprintingpagesfalse%
2468     \fi%
2469     \finish@Pages@notes%Needed to prevent final notes overlap line number
2470     \ignorespaces\fi}
2471
2472
2473 %
```

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.

```

2474 \newcommand{\finish@Pages@notes}{%
2475   \def\do##1{%
2476   %

```

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

```

2477   \ifnocritical@%
2478     \global\newnamebox{##1footins}
2479   \fi
2480   \ifnofamiliar@%
2481     \global\newnamebox{footins##1}
2482   \fi
2483 %

```

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

```

2484   \ifvoid\csuse{##1footins}%
2485     \ifvoid\csuse{footins##1}\else%
2486       \newpage\null%
2487       \listbreak%
2488     \fi%
2489   \else%
2490     \newpage\null%
2491     \listbreak%
2492   \fi%
2493 }%
2494 \dolistloop{\@series}%
2495 }%
2496 %

```

XIX.4 Struts

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

```

\ledstrutR
2497 \newcommand*{\ledstrutL}{}
2498 \newcommand*{\ledstrutR}{}
2499
2500 %

```

XIX.5 Page clearing

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

```

2501 \providecommand{\cleartoevenpage}[1][\@empty]{%
2502   \clearpage
2503   \ifodd\c@page\hbox{##1}\clearpage\fi}
2504
2505 \newcommand*{\cleartol@evenpage}{%

```

```

2506 \ifdim\pagetotal<\topskip% on an empty page
2507 \else
2508 \clearpage
2509 \Pages@mainmatter%
2510 \fi
2511 \ifodd\c@page%
2512 \ifprevpgnotnumbered%
2513 \addtocounter{par@page}{-1}%
2514 \ifdef{\prevpgstyle}{\thispagestyle{\prevpgstyle}}}%
2515 \fi%
2516 \hbox{}\clearpage%
2517 \fi%
2518 }%
2519 %

```

`\clearl@dleftpage` and `\clearl@drighthouse` 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`

```

2520 \newcommand*{\clearl@dleftpage}{%
2521 \ifdim\pagetotal=0pt\hbox{}\fi%
2522 \newpage%
2523 \ifodd\c@page\else
2524 \led@err@LeftOnRightPage
2525 \hbox{}\fi%
2526 \cleardoublepage
2527 \fi}
2528
2529 \newcommand*{\clearl@drighthouse}{%
2530 \ifdim\pagetotal=0pt\hbox{}\fi%
2531 \newpage%
2532 \ifodd\c@page
2533 \led@err@RightOnLeftPage
2534 \hbox{}\fi%
2535 \cleartoevenpage
2536 \fi}
2537
2538 %

```

XIX.6 Lines managing

`\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@linesinparL
2539 \newcommand*{\getlinesfromparlistL}{%

```

```

2540 \ifx\linesinpar@listL\empty
2541 \gdef\@cs@linesinparL{0}%
2542 \else
2543 \gl@p\linesinpar@listL\to\@cs@linesinparL
2544 \fi}
2545 \newcommand*\getlinesfromparlistR{%
2546 \ifx\linesinpar@listR\empty
2547 \gdef\@cs@linesinparR{0}%
2548 \else
2549 \gl@p\linesinpar@listR\to\@cs@linesinparR
2550 \fi}
2551
2552 %

```

`\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`.

```

2553 \newcommand*\getlinesfrompagelistL{%
2554 \ifx\linesonpage@listL\empty
2555 \gdef\@cs@linesonpageL{1000}%
2556 \else
2557 \gl@p\linesonpage@listL\to\@cs@linesonpageL
2558 \fi}
2559 \newcommand*\getlinesfrompagelistR{%
2560 \ifx\linesonpage@listR\empty
2561 \gdef\@cs@linesonpageR{1000}%
2562 \else
2563 \gl@p\linesonpage@listR\to\@cs@linesonpageR
2564 \fi}
2565
2566 %

```

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

```

2567 \newcommand*\@writelinesonpageL}[1]{%
2568 \edef\next{\write\linenum@out{\string\@lopL{#1}}}%
2569 \next}
2570 \newcommand*\@writelinesonpageR}[1]{%
2571 \edef\next{\write\linenum@outR{\string\@lopR{#1}}}%
2572 \next}
2573
2574 %

```

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

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

```

2575 \newcommand*{\l@dcalc@maxoftwo}[3]{%
2576   \ifnum #2>#1\relax
2577     #3=#2\relax
2578   \else
2579     #3=#1\relax
2580   \fi}
2581 \newcommand*{\l@dcalc@minoftwo}[3]{%
2582   \ifnum #2<#1\relax
2583     #3=#2\relax
2584   \else
2585     #3=#1\relax
2586   \fi}
2587
2588 %

```

XIX.7 Page break managing

`\ifl@dsamepage` `\checkpageL` tests if the space and lines already taken on the page by text and foot-
`\l@dsamepagetrue` notes is less than the constraints. If so, then `\ifl@dpagfull` is set FALSE and
`\l@dsamepagefalse` `\ifl@dsamepage` is set TRUE. If the page is spatially full then `\ifl@dpagfull` is set
`\ifl@dpagfull` TRUE and `\ifl@dsamepage` is set FALSE. If it is not spatially full but the maximum
`\l@dpagfulltrue` number of lines have been output then both `\ifl@dpagfull` and `\ifl@dsamepage`
`\l@dpagfullfalse` are set FALSE.

```

\checkpageL
\checkpageR
2589 \newif\ifl@dsamepage
2590 \l@dsamepagetrue
2591 \newif\ifl@dpagfull
2592
2593 \newcommand*{\checkpageL}{%
2594   \l@dpagfulltrue
2595   \l@dsamepagetrue
2596   \check@goal
2597   \ifdim\pagetotal<\ledthegoal
2598     \ifnum\numpagelinesL<\l@dmminpagelines
2599       \else
2600         \ifnomaxlines%
2601         \else%
2602           \l@dsamepagefalse%
2603           \l@dpagfullfalse%
2604         \fi%
2605       \fi
2606     \else
2607       \l@dsamepagefalse
2608       \l@dpagfulltrue
2609     \fi%
2610   \ifprint@last@after@pendL%
2611     \l@dpagfullfalse%
2612     \l@dsamepagefalse%
2613   \print@last@after@pendLfalse%

```

```

2614 \fi%
2615 }%
2616
2617 \newcommand*\checkpageR{%
2618 \l@dpagetrue
2619 \l@dsamepagetrue
2620 \check@goal
2621 \ifdim\pagetotal<\ledthegoal
2622 \ifnum\numpagelinesR<\l@dminpagelines
2623 \else
2624 \ifnomaxlines%
2625 \else%
2626 \l@dsamepagefalse%
2627 \l@dpagetruefalse%
2628 \fi%
2629 \fi
2630 \else
2631 \l@dsamepagefalse
2632 \l@dpagetrue
2633 \fi%
2634 \ifprint@last@after@pendR%
2635 \l@dpagetruefalse%
2636 \l@dsamepagefalse%
2637 \print@last@after@pendRfalse%
2638 \fi%
2639 }%
2640
2641 %

```

`\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`.

```

2642 \newcommand\checkpbL{
2643 \IfStrEq{\ledpb@setting}{after}{
2644 \xifinlistcs{\the\absline@num}{l@prev@pb}{\l@dpagetrue\
l@dsamepagefalse}{
2645 \xifinlistcs{\the\absline@num}{l@prev@nopb}{\l@dpagetruefalse\
l@dsamepagetrue}{
2646 }{
2647 \IfStrEq{\ledpb@setting}{before}{
2648 \numdef{\next@absline}{\the\absline@num+1}
2649 \xifinlistcs{\next@absline}{l@prev@pb}{\l@dpagetrue\
l@dsamepagefalse}{
2650 \xifinlistcs{\next@absline}{l@prev@nopb}{\l@dpagetruefalse\
l@dsamepagetrue}{
2651 }{
2652 }
2653
2654 \newcommand\checkpbR{
2655 \IfStrEq{\ledpb@setting}{after}{

```

```

2656 \xifinlistcs{\the\absline@numR}{l@prev@pbR}{\l@dpagfulltrue\
l@dsamepagefalse}{\}
2657 \xifinlistcs{\the\absline@numR}{l@prev@nopbR}{\l@dpagfullfalse\
l@dsamepagetrue}{\}
2658 }{\}
2659 \IfStrEq{\led@pb@setting}{before}{
2660 \numdef{\next@abslineR}{\the\absline@numR+1}
2661 \xifinlistcs{\next@abslineR}{l@prev@pbR}{\l@dpagfulltrue\
l@dsamepagefalse}{\}
2662 \xifinlistcs{\next@abslineR}{l@prev@nopbR}{\l@dpagfullfalse\
l@dsamepagetrue}{\}
2663 }{\}
2664 }
2665 %

```

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

\checkverseR

```

2666 \newcommand{\checkverseL}{
2667 \ifinstanzaL
2668 \iflednopbinverse
2669 \ifinserthangingsymbol
2670 \numgdef{\prev@abslineverse}{\the\absline@num-1}
2671 \IfStrEq{\led@pb@setting}{after}{\lednopbnum{\prev@abslineverse}}{\}
2672 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesL<3\ledpbnum{\
prev@abslineverse}\fi}{\}
2673 \fi
2674 \fi
2675 \fi
2676 }
2677 \newcommand{\checkverseR}{
2678 \ifinstanzaR
2679 \iflednopbinverse
2680 \ifinserthangingsymbolR
2681 \numgdef{\prev@abslineverse}{\the\absline@numR-1}
2682 \IfStrEq{\led@pb@setting}{after}{\lednopbnumR{\prev@abslineverse}}{\}
2683 \IfStrEq{\led@pb@setting}{before}{\ifnum\numpagelinesR<3\ledpbnumR{\
prev@abslineverse}\fi}{\}
2684 \fi
2685 \fi
2686 \fi
2687 }
2688 %

```

\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

```

2689 \newdimen\ledthegoal
2690 \ifshiftedpstarts

```

```

2691 \newcommand*{\@goalfraction}{0.95}
2692 \else
2693 \newcommand*{\@goalfraction}{0.9}
2694 \fi
2695
2696 \newcommand*{\check@goal}{%
2697 \ledthegoal=\@goalfraction\pagegoal}
2698 \newcommand{\setgoalfraction}[1]{%
2699 \xdef\@goalfraction{#1}%
2700 }
2701 %

```

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

```

\ifwrittenlinesL
2702 \newif\ifwrittenlinesL
2703 \newif\ifwrittenlinesR
2704
2705 %

```

XIX.8 Getting boxes content

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

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

```

2706 \newif\if@getnextbox%
2707 %

```

`\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.

```

2708 \newcommand*{\get@nextboxL}{%
2709 \ifvbox\namebox{1@dLcolrawbox\the\1@dpscL}% box is not empty
2710 %

```

The current box is not empty; do nothing.

```

2711 \else% box is empty
2712 %

```

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

```

2713 \@getnextboxtrue%Should be local, but be cautious
2714 %

```

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


```

2715 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscl}>\@donetotallinesL
2716 \parledgroup@notes@endL%
2717 \unless\ifnosyncpstarts%
2718 \getnextboxfalse%
2719 %

```

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

```

2720 \ifnomaxlines%
2721 \ifdim\pagetotal<\ledthegoal%
2722 \numdef{\@tmp}{\l@dpscl+1}%
2723 \ifcsdef{afterlines@pstart@\@tmp R}{%
2724 \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp R}}
2725 %
2726 {}%
2727 {\ifcsdef{minpage@pstart@\@tmp}%
2728 {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}}%
2729 {\ifnum\numpagelinesL=1@dminpagelines%
2730 \getnextboxtrue%
2731 \fi%
2732 }%
2733 {\@getnextboxtrue}}%
2734 {\@getnextboxtrue}}%
2735 }%
2736 {}%
2737 \fi%
2738 \fi%
2739 \fi%
2740 \else%
2741 \ifnomaxlines%
2742 \numdef{\@tmp}{\the\l@dpscl+1}%
2743 \ifcsdef{minpage@pstart@\@tmp}{%
2744 \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}}%
2745 {\ifdimgreater{\pagetotal}{\ledthegoal}%
2746 {\@getnextboxtrue}%
2747 {\@getnextboxfalse}%
2748 }%
2749 {\@getnextboxtrue}%
2750 }-}%
2751 \fi%
2752 \fi%
2753 %

```

Sufficient lines have been output.

```

2754 \if@getnextbox%
2755 \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscl}=\@donetotallinesL
2756 \parledgroup@notes@endL

```

```

2757 \fi
2758 \ifwrittenlinesL\else
2759 %

```

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

```

2760 \@writelinesinparL
2761 \writtenlinesLtrue
2762 \fi
2763 \ifnum\l@dnumstartsL>\l@dpscl
2764 %

```

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.

```

2765 \writtenlinesLfalse
2766 \ifbypstart@
2767 \global\line@num=0%
2768 \resetprevline%
2769 \fi
2770 % Add the content of the optional argument of the previous \protect\cs{pend
2771 }.
2772 % \begin{macrocode}
2773 \csuse{after@pendL@the\l@dpscl}%
2774 \global\csundef{after@pendL@the\l@dpscl}%
2775 %

```

Check the number of lines

```

2775 \l@dcalc@maxoftwo{\the\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2776 {\the\@donetotallinesL}%
2777 {\usernamecount{l@dmaxlinesinpar\the\l@dpscl}}%
2778 \global\@donetotallinesL \z@
2779 %

```

Go to the next pstart

```

2780 \global\advance\l@dpscl \@ne
2781 \global\pstartnumtrue%
2782 \restore@pstartL@pc%
2783 %

```

Add notes of parallel ledgroup.

```

2784 \parledgroup@notes@endL
2785 \parledgroup@correction@notes@spacing@final{L}
2786 \else
2787 %

```

```

2788 \print@last@after@pendLtrue%
2789 \fi
2790 \fi
2791 \fi}
2792 %

```

```

2793 \newcommand*{\get@nextboxR}{%
2794   \ifvbox\namebox{1@dRcolrawbox\the\l@dpscR}% box is not empty
2795   \else% box is empty
2796     \@getnextboxtrue%
2797     \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}>\@donetotallinesR
2798       \parledgroup@notes@endR
2799       \unless\ifnosyncpstarts%
2800         \@getnextboxfalse%
2801         \ifnomaxlines%
2802           \ifdim\pagetotal<\ledthegoal%
2803             \numdef{\@tmp}{\l@dpscR+1}%
2804             \ifcsdef{afterlines@pstart@\@tmp L}{%
2805               \ifnumless{\numpagelinesL}{\csuse{afterlines@pstart@\@tmp L}}
2806               {}%
2807               {\ifcsdef{minpage@pstart@\@tmp}%
2808                 {\ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2809                   {\ifnum\numpagelinesR=1@dminpagelines%
2810                     \@getnextboxtrue%
2811                     \fi%
2812                     }%
2813                     {\@getnextboxtrue}}}%
2814                   {\@getnextboxtrue}}}%
2815                 }%}%
2816               }%
2817             {}%
2818           \fi%
2819         \fi%
2820       \fi%
2821     \else%
2822       \ifnomaxlines%
2823         \numdef{\@tmp}{\the\l@dpscR+1}%
2824         \ifcsdef{minpage@pstart@\@tmp}{%
2825           \ifnumless{\the\c@page}{\csuse{minpage@pstart@\@tmp}}%
2826             {\ifdimgreater{\pagetotal}{\ledthegoal}%
2827               {\@getnextboxtrue}%
2828               {\@getnextboxfalse}%
2829             }%
2830             {\@getnextboxtrue}%
2831           }{}
2832         \fi%
2833       \fi%
2834     \if@getnextbox%
2835       \ifnum\usernamecount{1@dmaxlinesinpar\the\l@dpscR}=\@donetotallinesR
2836         \parledgroup@notes@endR
2837         \fi
2838       \ifwrittenlinesR\else
2839         \@writelinesinparR
2840         \writtenlinesRtrue
2841       \fi

```

```

2842 \ifnum\l@dnumstart>\l@dpscR
2843 \writtenlinesRfalse
2844 \ifbypstart@R
2845 \global\line@numR=0%
2846 \resetprevline@%
2847 \fi
2848 \csuse{after@pendR@the\l@dpscR}%
2849 \global\csundef{after@pendR@the\l@dpscR}%
2850 \l@dcalc@maxoftwo{\the\usernamecount{l@maxlinesinpar\the\l@dpscR}}%
2851 \the\@donetotallinesR}%
2852 \usernamecount{l@maxlinesinpar\the\l@dpscR}}%
2853 \global\@donetotallinesR \z@
2854 \global\advance\l@dpscR \@ne
2855 \global\pstartnumRtrue%
2856 \restore@pstartR@pc%
2857 \parledgroup@notes@endR
2858 \parledgroup@correction@notes@spacing@final{R}
2859 \else
2860 \print@last@after@pendRtrue%
2861 \fi
2862 \fi
2863 \fi}
2864
2865 %

```

XX Page numbering

XX.1 Global options

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

We cannot implement these two options by changing the value of the page counter, since its value is used by many \LaTeX 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 \LaTeX '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 \LaTeX 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.

```

2866
2867 \newcommand{\par@patch@thepage}{%
2868   \ifboolexpr{%
2869     bool{sameparallelnumber}%
2870     or bool{prevpgnotnumbered}%
2871   }%
2872   {%
2873     \patchcmd{\thepage}%
2874       {page}{par@page}%
2875       {}%
2876       {\led@error@fail@patch@thepage}%
2877   }{}%
2878 }%
2879
2880 \newcommand{\par@patch@pagenumbering}{%
2881   \ifboolexpr{%
2882     bool{sameparallelnumber}%
2883     or bool{prevpgnotnumbered}%
2884   }%
2885   {%
2886     \setcounter{par@page}{1}%
2887   }%
2888   {}%
2889   \par@patch@thepage%
2890 }%
2891
2892 \ifl@dmemoir%
2893   \apptocmd{\@mempnum}%
2894     {\par@patch@pagenumbering}%
2895     {}%
2896     {\led@error@fail@patch@mempnum}%
2897
2898 \else%
2899   \apptocmd{\pagenumbering}%
2900     {\par@patch@pagenumbering}%
2901     {}%
2902     {\led@error@fail@patch@pagenumbering}%
2903 \fi%
2904
2905 \AtBeginDocument{\par@patch@thepage}%
2906 %

```

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

```

2907 \AtBeginDocument{%

```

```

2908 \apptocmd{\@outputpage}{%
2909   \ifsameparallelpagenumber%
2910     \ifl@dprintingpages%
2911       \ifodd\c@page\else%
2912         \stepcounter{par@page}%
2913       \fi%
2914     \else%
2915       \stepcounter{par@page}%
2916     \fi%
2917   \else%
2918     \stepcounter{par@page}%
2919   \fi%
2920 }%
2921 {}%
2922 {\led@error@fail@patch@@@outputpage}%
2923 }
2924 %

```

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

```

2925 \newcounter{par@page}%
2926 \setcounter{par@page}{1}%
2927 %

```

XX.2 mainmatter option of \Pages

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

```

\ifPages@mainmatter 2928 \newif\ifPages@mainmatter
\Pages@mainmatter 2929 \newcommand{\Pages@mainmatter}{%
2930   \ifPages@mainmatter%
2931     \pagenumbering{arabic}%
2932     \addtocounter{page}{1}%
2933     \addtocounter{par@page}{-1}%
2934     \patchcmd{\thepage}{page}{par@page}{}{}%
2935   \fi%
2936 }
2937 %

```

XXI Sections' titles' commands

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

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

```

\chapter 2938 \pretocmd{\chapter}{%
2939 \ifl@dprintingpages%
2940 \vbox{}}%
2941 \fi%
2942 }%
2943 {}%
2944 {}%
2945 %

```

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

```

2946 \newcommand{\eledsectnotoc}[1]{\xdef\@eledsectnotoc{#1}}
2947 \eledsectnotoc{R}
2948 %

```

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

```

2949 \newcommand{\eledsectmark}[1]{\xdef\@eledsectmark{#1}}
2950 \eledsectmark{L}
2951 %

```

\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.

```

2952 \newskip\eledsection@correcting@skip
2953 %

```

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

```

2954 \newwrite\eled@sectioningR@out
2955 %

```

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).

```

2956 \def\l@prev@pbR{}
2957 \def\l@prev@nopbR{}
2958 %

```

`\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.

```
2959 \newcommand{\ledpbR}{\write\linenum@outR{\string\led@pbR}}
2960 \newcommand{\ledpbnumR}[1]{\write\linenum@outR{\string\led@pbnumR{#1}}}
2961 \newcommand{\lednopbR}{\write\linenum@outR{\string\led@nopbR}}
2962 \newcommand{\lednopbnumR}[1]{\write\linenum@outR{\string\led@nopbnumR{#1}}}
2963 %
```

`\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.

```
2964 \newcommand{\led@pbR}{\listxadd{\l@prev@pbR}{\the\absline@numR}}
2965 \newcommand{\led@pbnumR}[1]{\listxadd{\l@prev@pbR}{#1}}
2966 \newcommand{\led@nopbR}{\listxadd{\l@prev@nopbR}{\the\absline@numR}}
2967 \newcommand{\led@nopbnumR}[1]{\listxadd{\l@prev@nopbR}{#1}}
2968 %
```

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).

```
2969 \newmarks\parledgroup@
2970 \newmarks\parledgroup@series
2971 \newmarks\parledgroup@type
2972 %
```

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

```
2973 \newcommand{\parledgroup@notes@startL}{%
2974   \ifnum\usenamecount{1@dmxlinesinpar\the\l@dpscL}>0%
2975   \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{}%
2976   \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{}%
2977   \fi%
2978   \global\ledgroupnotesL@true%
2979   \insert@noterule@ledgroup{L}%
2980 }
2981 \newcommand{\parledgroup@notes@startR}{%
2982   \ifnum\usenamecount{1@dmxlinesinpar\the\l@dpscR}>0%
```



```

2983 \IfStrEq{\splitfirstmarks\parledgroup@type}{footnoteX}{\csuse{
bhooknoteX@\splitfirstmarks\parledgroup@series}}{ }%
2984 \IfStrEq{\splitfirstmarks\parledgroup@type}{Xfootnote}{\csuse{
bhookXnote@\splitfirstmarks\parledgroup@series}}{ }%
2985 \fi%
2986 \global\ledgroupnotesR@true%
2987 \insert@noterule@ledgroup{R}%
2988 }
2989 %

```

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

```

2990 \newcommand{\parledgroup@notes@endL}{%
2991 \global\ledgroupnotesL@false%
2992 }
2993 \newcommand{\parledgroup@notes@endR}{%
2994 \global\ledgroupnotesR@false%
2995 }
2996 %

```

`\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`

```

2997 \newcommand{\insert@noterule@ledgroup}[1]{
2998 \IfStrEq{\splitbotmarks\parledgroup@}{begin}{%
2999 \IfStrEq{\splitbotmarks\parledgroup@type}{Xfootnote}{
3000 \csuse{ifledgroupnotes#1@}
3001 \vskip\skip\csuse{mp\splitbotmarks\parledgroup@series footins}
3002 \csuse{\splitbotmarks\parledgroup@series footnoterule}
3003 \fi
3004 }
3005 {}
3006 \IfStrEq{\splitbotmarks\parledgroup@type}{footnoteX}{
3007 \csuse{ifledgroupnotes#1@}
3008 \vskip\skip\csuse{mpfootins\splitbotmarks\parledgroup@series}
3009 \csuse{footnoterule\splitbotmarks\parledgroup@series}
3010 \fi
3011 }{}
3012 }
3013 {}
3014 }
3015 %

```

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

```

3016 \newcommand{\setparledgroupnotespadding}[1]{\gdef\@parledgroupnotespadding
{#1}}
3017 \newcommand{\@parledgroupnotespadding}{}
3018 %

```

`\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`.

```

3019 \dimdef{\parledgroup@notespacing@correction}{0pt}
3020 \newcommand{\parledgroup@notespacing@set@correction}{%
3021   {\@getfirstseries\csuse{Xnotefontsize@\@firstseries}%We suppose all the
series has the same footnote size setup
3022   \@parledgroupnotespacing\dimgdef{\temp@spacing}{\baselineskip}}%
3023   \dimgdef{\parledgroup@notespacing@correction}{\baselineskip-\temp@spacing}
}%
3024 }
3025 %

```

`\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.

```

3026 \newcommand{\parledgroup@correction@notespacing@init}{
3027   \dimdef{\parledgroup@notespacing@correction@accumulated}{0pt}
3028   \dimdef{\parledgroup@notespacing@correction@modulo}{0pt}
3029 }
3030 \parledgroup@correction@notespacing@init
3031 %

```

`\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.

```

3032 \newcommand{\parledgroup@correction@notespacing@final}[1]{
3033   \ifparledgroup
3034     \vspace{\parledgroup@notespacing@correction@accumulated}
3035     \parledgroup@correction@notespacing@init%
3036     \ifstrequal{#1}{L}{
3037       \numdef{\@checking}{\the\l@dpscl-1}
3038     }{
3039       \numdef{\@checking}{\the\l@dpscr-1}
3040     }
3041     \dimdef{\@beforenotes@current@diff}{\csuse{\@parledgroup@beforenotes@\@checking L}-\csuse{\@parledgroup@beforenotes@\@checking R}}%
3042     \ifstrequal{#1}{L}%
3043     {% Left
3044       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{-\@beforenotes@current@diff}}%
3045     }%
3046     {% Right
3047       \ifdimgreater{\@beforenotes@current@diff}{0pt}{\vspace{\@beforenotes@current@diff}}%
3048     }%

```

```

3049 \fi
3050 }
3051 %

```

`\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.

```

3052 \newcommand{\parledgroup@correction@notespacing}[1]{%
3053   \csuse{ifledgroupnotes#1@}%
3054   \vspace{-\parledgroup@notespacing@correction}%
3055   \dimdef{\parledgroup@notespacing@correction@accumulated}{\
parledgroup@notespacing@correction@accumulated+
parledgroup@notespacing@correction}%
3056   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo+
parledgroup@notespacing@correction}%
3057   \ifdimless{\parledgroup@notespacing@correction@modulo}{\baselineskip
}{\advance\numpagelinesL -\@ne%
3058   \dimdef{\parledgroup@notespacing@correction@modulo}{\
parledgroup@notespacing@correction@modulo-\baselineskip}%
3059   }% mean greater than equal
3060   \fi%
3061 }
3062 %

```

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

```

3063 \dimdef\parledgroup@beforenotesL{0pt}
3064 \dimdef\parledgroup@beforenotesR{0pt}
3065 %

```

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

```

3066 \newcommand{\parledgroup@beforenotes@save}[1]{
3067   \ifparledgroup
3068     \csdimgdef{@parledgroup@beforenotes@\the\csuse{1@dnumpstarts#1}#1}{\
csuse{\parledgroup@beforenotes#1}}

```

```
3069 \csdimgdef{parledgroup@beforenotes#1}{0pt}  
3070 \fi  
3071 }  
3072 %
```

XXIV Compatibility with eledmac

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

```
3073 \ifeledmaccompat@%  
3074  
3075  
3076 \unless\ifnocritical@  
3077 \let\onlyXside\Xonlyside  
3078 \fi  
3079 \fi  
3080 %
```

XXV The End

</code>

Appendix A Some things to do when changing version

Appendix A.1 Migration to eledpar 1.4.3

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

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

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

See the following two examples:

With standard `\hangingsymbol`:

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

With the modification of the `hangingsymbol`:

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

Appendix A.2 Migration from eledpar to reledpar

As for migration from eledmac to reledmac:

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

Appendix A.2.1 Deprecated options

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

Appendix A.2.2 `\renewcommand` replaced with command

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

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

Appendix A.2.3 Commands the names of which have changed

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

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

Appendix A.3 Migration to `reledpar` 2.2.0

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

Appendix A.4 Migration to `reledpar` 2.3.0

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

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

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

Appendix A.5 Migration to `reledpar` 2.4.0

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

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

Appendix A.6 Migration to `reledpar` 2.5.0

If you use `\stanza` or `astanza` environment, please read Appendix A.12 p. 309.

Appendix A.7 Migration to `reledpar` 2.6.0

`\printlinenumR` was deleted. Use `\Xlineflag` instead.

Appendix A.8 Migration to `reledpar` 2.6.1

If you use `perpage` package to control footnote numbering, please read the handbook on 5.3.3 p. 13.

References

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

Index

Symbols

<code>\@adv</code>	1
<code>\@astanza@line</code>	1
<code>\@cs@linesinparL</code>	1
<code>\@cs@linesinparR</code>	1
<code>\@cs@linesonpageL</code>	1
<code>\@cs@linesonpageR</code>	1
<code>\@donereallinesL</code>	1
<code>\@donereallinesR</code>	1
<code>\@donetotallinesL</code>	1
<code>\@donetotallinesR</code>	1
<code>\@eledsectionL</code>	1
<code>\@eledsectionR</code>	1
<code>\@lab</code>	1
<code>\@lopL</code>	1
<code>\@lopR</code>	1
<code>\@nl</code>	1
<code>\@nl@regR</code>	1
<code>\@outputpage</code>	1
<code>\@par@sync@option</code>	1
<code>\@par@this@sync@option</code>	1
<code>\@parledgroupnotespacing</code>	1
<code>\@pend</code>	1
<code>\@pendR</code>	1
<code>\@pstart</code>	1
<code>\@pstartR</code>	1
<code>\@pstartsfalse</code>	1
<code>\@pstartstrue</code>	1
<code>\@ref</code>	1
<code>\@ref@regR</code>	1
<code>\@set</code>	1
<code>\@stopastanza</code>	1
<code>\@writelinesinparL</code>	1

\@writelinesinparR	1
\@writelinesonpageL	1
\@writelinesonpageR	1
\@writepageofparL	1
\@writepageofparR	1
CLASSmemoir	108
COMMAND\+	46
COMMAND\@Rlineflag	73, 144
COMMAND\@adv	37, 142
COMMAND\@cs@linesinparL	99
COMMAND\@cs@linesonpageL	100
COMMAND\@eledsectionL	83
COMMAND\@eledsectionR	83
COMMAND\@eledsectmark	111
COMMAND\@eledsectnotoc	111
COMMAND\@footnotetext	75
COMMAND\@goalfraction	12, 103
COMMAND\@l@dtempcnta	67
COMMAND\@lab	36, 73, 142
COMMAND\@lopL	41, 94, 100
COMMAND\@lopR	41, 100
COMMAND\@mempnum	108
COMMAND\@namedef	77
COMMAND\@namuse	77
COMMAND\@nl	36, 42, 43, 73, 142
COMMAND\@nl@regR	36
COMMAND\@outputpage	108, 109
COMMAND\@page	73
COMMAND\@par@sync@option	35
COMMAND\@parledgroupnotespacing	113
COMMAND\@pend	41
COMMAND\@pendR	41
COMMAND\@pstart	41
COMMAND\@pstartstrue	81
COMMAND\@ref	39, 40, 43, 142
COMMAND\@ref@regR	39
COMMAND\@set	37, 142
COMMAND\@sw	40
COMMAND\AtBeginPairs	8, 49, 140
COMMAND\AtEveryPend	140–142
COMMAND\AtEveryPstart	2, 17, 55, 84, 140–142
COMMAND\AtEveryPstartCall	2, 17, 55, 141
COMMAND\Clear the right lines for \read@linelist	35
COMMAND\Columns	7, 8, 17, 47, 83, 87, 137–140, 142, 143
COMMAND\Columns@print@after@pend	90
COMMAND\Columns@print@before@pstart	90
COMMAND\Lcolwidth	8, 10, 92
COMMAND\Leftsidehook	137
COMMAND\Leftsidehookend	137
COMMAND\MakePerPage	13

COMMAND\Pages	4, 7, 10, 12, 17, 46, 47, 65, 68, 70, 91, 92, 97, 108, 110, 114, 137, 140–143
COMMAND\Pages@mainmatter	110
COMMAND\Pairs	46
COMMAND\Rcolwidth	8, 10, 92
COMMAND\Rightsidehook	137
COMMAND\Rightsidehookend	137
COMMAND\Rlineflag	117
COMMAND\Xendlineflag	14, 143
COMMAND\Xlineflag	14, 118, 143
COMMAND\Xmaxhnotes	13
COMMAND\Xnoteswidthliketwocolumns	9, 140
COMMAND\Xonlyside	14, 68, 118
COMMAND\&	18
COMMAND\absline@numR	34
COMMAND\add@penalties	67
COMMAND\add@penaltiesL	67
COMMAND\advanceline	37, 43, 142
COMMAND\affixline@num	62
COMMAND\affixline@numR	62, 137, 138
COMMAND\affixpstart@num	65
COMMAND\affixpstart@numR	65
COMMAND\affixside@note	74
COMMAND\aftercolumnseparator	9, 87, 139
COMMAND\araw@textfalse	82
COMMAND\araw@texttrue	82
COMMAND\at@begin@pairs	49
COMMAND\autopar	17
COMMAND\ballast@count	67
COMMAND\baselineskip	87, 115
COMMAND\bbl@set@language	78, 79, 142
COMMAND\beforecolumnseparator	9, 87, 139
COMMAND\begin	18
COMMAND\beginnumbering	15–17, 26, 36, 87, 138, 139, 142
COMMAND\beginnumberingR	42
COMMAND\bf	138
COMMAND\bfseries	138
COMMAND\brokenpenalty	67
COMMAND\chapter	110, 137
COMMAND\check@goal	103
COMMAND\check@pstarts	81
COMMAND\checkpageL	94, 101
COMMAND\checkpb@columns	86
COMMAND\checkpbL	102
COMMAND\checkpbR	102
COMMAND\checkraw@text	82
COMMAND\checkverseL	103
COMMAND\checkverseR	103
COMMAND\clear(double)page	98
COMMAND\clearl@dleftpage	99
COMMAND\clearl@drightpage	99

COMMAND\clearpage	99, 141
COMMAND\cleartoevenpage	98
COMMAND\cleartol@devenpage	98, 110
COMMAND\columnrulewidth	9, 87
COMMAND\columns@position	87
COMMAND\columnseparator	9
COMMAND\columnspan	9, 139
COMMAND\correct@Xfootins@box	70, 71, 141
COMMAND\correct@footinsX@box	70, 141
COMMAND\critext	141
COMMAND\csname	48
COMMAND\displaywidowpenalty	67
COMMAND\do@actions	61
COMMAND\do@actions@fixedcode	137
COMMAND\do@actions@nextR	61
COMMAND\do@actionsR	61, 137
COMMAND\do@ballast	67
COMMAND\do@ballastR	61
COMMAND\do@insidelineLhook	139
COMMAND\do@insidelineRhook	139
COMMAND\do@line	56
COMMAND\do@line(L/R)	58
COMMAND\do@lineL	56, 67, 137, 138
COMMAND\do@lineLhook	137
COMMAND\do@lineR	59, 137–139
COMMAND\do@lineRhook	137
COMMAND\do@lockoff	143
COMMAND\do@lockoffR	38
COMMAND\do@lockon	142
COMMAND\do@lockonR	37
COMMAND\doinsidelineLhook	140
COMMAND\doinsidelineRhook	140
COMMAND\dolineLhook	140
COMMAND\dolineRhook	140
COMMAND\edindex	141
COMMAND\edlabel	138, 141
COMMAND\edtext	39, 43, 140, 141
COMMAND\eled@sectioningR@out	111
COMMAND\eledchapter	141
COMMAND\eledsection	140, 141, 143
COMMAND\eledsection@correcting@skip	111
COMMAND\eledsectmark	20, 111
COMMAND\eledsectnotoc	20, 111
COMMAND\eledxxx	140
COMMAND\end	18
COMMAND\endgraf	54
COMMAND\endlock	43, 142
COMMAND\endnumbering	15, 17, 27, 142
COMMAND\endsub	43, 142
COMMAND\endumbering	15

COMMAND\expandafter	45
COMMAND\extensionchars	26
COMMAND\firstlinenum	16, 139, 143
COMMAND\firstsublinenum	139, 143
COMMAND\fix@page	37, 142
COMMAND\flag@end	43, 137, 140
COMMAND\flag@start	43, 140
COMMAND\flush@notesR	68
COMMAND\footnote	47
COMMAND\footnoteX	44, 45
COMMAND\footnoteXmk	14
COMMAND\footnoteXnomk	14, 45
COMMAND\frontmatter	12, 20
COMMAND\get@nextboxL	138
COMMAND\get@nextboxR	138
COMMAND\getline@numL	60
COMMAND\getline@numR	60
COMMAND\getlinesfrompagelistL	100
COMMAND\getlinesfrompagelistR	100
COMMAND\getlinesfromparlistL	99
COMMAND\getlinesfromparlistR	99
COMMAND\gl@p	45
COMMAND\goalfraction	117
COMMAND\hangingsymbol	117, 138
COMMAND\hfill	86, 87
COMMAND\hidenumbering	16, 142
COMMAND\if@getnextbox	104
COMMAND\ifPages@mainmatter	110
COMMAND\ifbypage@	142
COMMAND\ifbypstart@R	142
COMMAND\ifdim	86
COMMAND\ifinserthangingsymbol	75
COMMAND\ifinserthangingsymbolR	75
COMMAND\ifl@dpagefull	101
COMMAND\ifl@dpaging	23, 140
COMMAND\ifl@dpairing	23, 137
COMMAND\ifl@dsamelang	139
COMMAND\ifl@dsamepage	101
COMMAND\ifl@pagefull	94
COMMAND\ifledRcol	23
COMMAND\iflledRcol	138
COMMAND\ifnumberedpar@	51
COMMAND\ifnumberingR	138
COMMAND\ifnumberpstart	47
COMMAND\ifpst@rtedL	26, 27, 52, 137
COMMAND\ifpst@rtedR	26
COMMAND\ifsublines@	37
COMMAND\insert@countR	39
COMMAND\insert@noterule@ledgroup	113
COMMAND\insertlines@list	39

COMMAND\insertlines@listR	39
COMMAND\inserts@list	51
COMMAND\inserts@listR	66
COMMAND\l@d@nums	43
COMMAND\l@d@set	37, 43, 142
COMMAND\l@dLcolrawbox	51
COMMAND\l@dLcolrawbox1	80
COMMAND\l@dLcolrawbox2	80
COMMAND\l@dRcolrawbox	51
COMMAND\l@dbfnote	75, 142
COMMAND\l@dcalc@maxoftwo	100
COMMAND\l@dcalc@minoftwo	100
COMMAND\l@dchecklang	137, 140
COMMAND\l@dcsnote	139
COMMAND\l@dleftbox	56, 94, 141
COMMAND\l@dlinenumR	34, 137
COMMAND\l@dlsnote	139
COMMAND\l@dmake@labels	73
COMMAND\l@dmaxlinesinpar	92
COMMAND\l@dmaxlinesinpar1	80
COMMAND\l@dminpagelines	93, 137
COMMAND\l@dnumstartL	80, 137
COMMAND\l@dprintingcolumnstrue	141
COMMAND\l@dprintingpagestrue	141
COMMAND\l@dpscL	84, 92, 93, 106
COMMAND\l@dpscR	84, 92, 93
COMMAND\l@drsnote	139
COMMAND\l@dsetupmaxlinecounts	80
COMMAND\l@duselanguage	78, 79, 137
COMMAND\l@dzeromaxlinecounts	80
COMMAND\l@prev@nopbR	111
COMMAND\l@prev@pbR	111
COMMAND\labelstarttrue	138
COMMAND\labelref@list	73
COMMAND\labelref@listR	73
COMMAND\lang	78
COMMAND\last@page@numR	37
COMMAND\led	138
COMMAND\led@nopbR	112
COMMAND\led@nopbnumR	112
COMMAND\led@pbR	112
COMMAND\led@pbnumR	112
COMMAND\ledinnerrote	19
COMMAND\ledleftnote	19
COMMAND\lednopb	18, 86, 102
COMMAND\lednopbR	112
COMMAND\lednopbnumR	112
COMMAND\ledouterote	19
COMMAND\ledpb	86, 102
COMMAND\ledpbR	112

COMMAND\ledpbnumR	112
COMMAND\ledrightnote	19
COMMAND\ledsidenote	19
COMMAND\ledstrutL	137
COMMAND\ledstrutR	137, 143
COMMAND\ledthegoal	103
COMMAND\ledtrutL	137, 143
COMMAND\leftlinenumR	34, 137
COMMAND\let	45
COMMAND\line@list@R	40
COMMAND\line@list@stuff	36, 42
COMMAND\line@margin	31
COMMAND\line@marginR	31, 137
COMMAND\line@numR	34
COMMAND\lineation	16, 141
COMMAND\lineation*	16, 30, 140
COMMAND\lineationR	16, 30, 141
COMMAND\linenum@out	73
COMMAND\linenum@outR	42
COMMAND\linenumberstyle	16, 118
COMMAND\linenumberstyle*	118
COMMAND\linenumberstyleR	16, 118
COMMAND\linenumincrement	16, 139, 143
COMMAND\linenummargin	16, 31, 137, 142, 143
COMMAND\linenummargin*	16, 31, 143
COMMAND\linenummarginR	16, 31, 143
COMMAND\linenumrepR	33, 137
COMMAND\linesinpar@listL	41, 99
COMMAND\linesonpage@listL	41, 100
COMMAND\lock@off	38
COMMAND\lock@on	37
COMMAND\mainmatter	2, 12, 20, 143
COMMAND\makeatletter	58
COMMAND\maxchunks	7, 18, 80, 81
COMMAND\maxhnotesX	13
COMMAND\memorydump	15, 29
COMMAND\n@num	141
COMMAND\new@lineL	42
COMMAND\new@lineR	43
COMMAND\newhookcommand@series	46
COMMAND\newif	141
COMMAND\newpage	98, 99, 141
COMMAND\newseries	47
COMMAND\newseries@	43
COMMAND\newseries@par	43, 46, 47
COMMAND\noeledxxx	140
COMMAND\nomark@	45
COMMAND\nomaxlines	41
COMMAND\normalbfnoteX	137, 142
COMMAND\notesXwidthliketwocolumns	9, 140

COMMAND\num@lines	67
COMMAND\num@lines(R)	51
COMMAND\numberingR	28
COMMAND\numberlinefalse	7
COMMAND\numberonlyfirstinline	138
COMMAND\numberpstartfalse	16
COMMAND\numberpstarttrue	16, 138, 143
COMMAND\one@line	51, 75
COMMAND\one@lineR	51
COMMAND\onlyXside	118
COMMAND\onlysideX	14, 68, 142
COMMAND\otherlanguage	143
COMMAND\page@action	37, 142
COMMAND\pagenumbering	108, 143
COMMAND\pages	12
COMMAND\pagetotal	94, 141
COMMAND\par@line	67
COMMAND\par@line(R)	51
COMMAND\par@patch@pagenumbering	109
COMMAND\par@patch@thepage	108, 109
COMMAND\par@sync@option	22
COMMAND\parledgroup@	112
COMMAND\parledgroup@beforenotes@save	115
COMMAND\parledgroup@beforenotesL	115
COMMAND\parledgroup@beforenotesR	115
COMMAND\parledgroup@correction@notespacing	115
COMMAND\parledgroup@correction@notespacing@final	114
COMMAND\parledgroup@correction@notespacing@init	114
COMMAND\parledgroup@notes@endL	113
COMMAND\parledgroup@notes@endR	113
COMMAND\parledgroup@notes@startL	112
COMMAND\parledgroup@notes@startR	112
COMMAND\parledgroup@notespacing@correction	114, 115
COMMAND\parledgroup@notespacing@correction@accumulated	115
COMMAND\parledgroup@notespacing@correction@modulo	115
COMMAND\parledgroup@notespacing@set@correction	114
COMMAND\parledgroup@series	112
COMMAND\parledgroup@type	112
COMMAND\parledgroupnotespacing	117
COMMAND\parledgrouptrue	19
COMMAND\patchcmd	142
COMMAND\pausenumbering	29
COMMAND\pend	3, 7, 10, 17–19, 47, 51, 54, 55, 80, 90, 139, 140, 142, 143
COMMAND\pendL	139, 140
COMMAND\pendR	140
COMMAND\pends	17
COMMAND\perpage	13
COMMAND\prev@nopbR	112
COMMAND\prev@pbR	112
COMMAND\prevpgstyle	23

COMMAND\print@Xnotes	68
COMMAND\print@Xnotes@forpages	68, 141
COMMAND\print@columnseparator	86, 140
COMMAND\print@eledsectionL	58
COMMAND\print@line	57
COMMAND\print@lineL	57
COMMAND\print@notesX@forpages	141
COMMAND\printlinenumR	118
COMMAND\printlinesR	143
COMMAND\pstart	3, 7, 10, 16–19, 30, 43, 47, 51, 52, 54, 55, 80, 84, 90, 138, 139, 142, 143
COMMAND\pstartL	55, 139
COMMAND\pstartR	55, 138, 139
COMMAND\pstartinfootnote	141
COMMAND\raw@text	80
COMMAND\read@linelist	35, 36, 120, 142
COMMAND\ref@reg	39
COMMAND\ref@regR	39, 142
COMMAND\relax	118
COMMAND\reledmac	142
COMMAND\renewcommand	117
COMMAND\resumenumbering	29, 139
COMMAND\resumenumberingR	140
COMMAND\rightlinenumR	34, 137
COMMAND\section	137
COMMAND\section@num	26
COMMAND\selectlanguage	17, 78, 79
COMMAND\set@line	43, 142
COMMAND\set@line@action	37, 142
COMMAND\setRlineflag	17, 117
COMMAND\setgoalfraction	12, 117
COMMAND\sethangingsymbol	18
COMMAND\setline	37, 43, 142
COMMAND\setlinenum	37, 43, 142
COMMAND\setnoteposition...	87
COMMAND\setparledgroupnotespacing	117, 143
COMMAND\setposition...	87
COMMAND\setstanzaindents	9, 18, 117
COMMAND\setwidth...	87
COMMAND\sidenotemargin	19, 140
COMMAND\sidenotemargin*	19, 140
COMMAND\skipnumbering	16, 141
COMMAND\sloppy	8
COMMAND\stanza	7, 9, 16, 18, 49, 75, 118, 138
COMMAND\stanzanumtrue	18
COMMAND\startlock	43, 142
COMMAND\startsub	43, 142
COMMAND\sub@action	37, 143
COMMAND\sub@off	73
COMMAND\sub@on	73
COMMAND\subline@numR	34

COMMAND\sublinenumberstyle	16, 118
COMMAND\sublinenumberstyle*	118
COMMAND\sublinenumberstyleR	16, 118
COMMAND\sublinenumincrement	139, 143
COMMAND\sublinenumrepR	33, 137
COMMAND\syntaxonly	144
COMMAND\sza@0@	18
COMMAND\textheight	13
COMMAND\textwidth	49
COMMAND\the@labelX	144
COMMAND\thefootnoteX	139
COMMAND\theledlanguageL	78, 79
COMMAND\theledlanguageR	78, 79
COMMAND\thepage	20, 108
COMMAND\thepstartL	16, 138
COMMAND\thepstartR	16, 138
COMMAND\thestanzaL	18
COMMAND\thestanzaR	18
COMMAND\ vbox	52
COMMAND\vl@dbfnote	75
COMMAND\vskip	113
COMMAND\vsplit	67
COMMAND\widthliketwocolumns	9
COMMAND\widthliketwocolumnsfalse	9
COMMAND\widthliketwocolumnstrue	9
COMMAND\xflagref	144
COMMAND\xright@appenditem	45
COMMAND\xspace	21
COMMAND\xxxfootstart	87
COMMAND\xxxmatter	108
ENVIRONMENTLeftside	49
ENVIRONMENTRightside	50
ENVIRONMENTastanza	18, 75, 76, 118, 143
ENVIRONMENTcolumns	28, 143
ENVIRONMENTledgroup	6
ENVIRONMENTleft	16
ENVIRONMENTpages	28, 48, 143
ENVIRONMENTpairs	48, 143
PACKAGEEDMAC	119
PACKAGEEDSTANZA	119
PACKAGEEledmac	44, 141
PACKAGEEledpar	141
PACKAGETABMAC	119
PACKAGEbabel	17, 78, 79, 143
PACKAGEedmac	119
PACKAGEeledmac	4, 80, 116, 117, 119, 139, 140, 142
PACKAGEeledpar	5, 6, 13, 32, 117, 118, 139–141
PACKAGEetoolbox	86, 111
PACKAGEledmac	6
PACKAGEledpar	1, 6

PACKAGEmemoir	119
PACKAGEMusixtex	139
PACKAGEperpage	2, 13, 118, 144
PACKAGEpolyglossia	17, 78, 79
PACKAGEreledmac	1, 3, 5–9, 13, 16–21, 23, 26, 27, 31, 32, 34, 36–39, 41–43, 45–47, 57, 65, 73, 75, 92, 98, 117, 118, 142, 143
PACKAGEreledpar	1, 3, 5–12, 18–23, 30, 34, 35, 41, 43, 44, 46, 47, 73, 117, 118, 142
PACKAGEsetspace	2, 20
PACKAGESyntonly	144
PACKAGEXkeyval	21

A

\absline@numR	1
\actionlines@listR	1
\actions@listR	1
\add@inserts@nextR	1
\add@insertsR	1
\add@penaltiesL	1
\add@penaltiesR	1
\advanceline	1
\affixline@numR	1
\affixpstart@numL	1
\affixpstart@numR	1
\affixside@noteR	1
\aftercolumnseparator	1, 9
\araw@textfalse	1
\araw@texttrue	1
astanza (environment)	18
\AtBeginPairs	1, 8
\AtEveryPstartCall	1
\autopar	17

B

\bbl@set@language	1
\beforecolumnseparator	1, 9
\beginnumbering	15
\beginnumberingR	1

C

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

<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@leftpage</code>	1
<code>\clearl@rightpage</code>	1
<code>\cleartoevenpage</code>	1
<code>\cleartol@evenpage</code>	1
<code>\columnrulewidth</code>	1, 9
<code>\Columns</code>	1, 8
<code>\columns@position</code>	1
<code>\Columns@print@after@pend</code>	1
<code>\Columns@print@before@pstart</code>	1
<code>\columnseparator</code>	1, 9
<code>\columnspostion</code>	1, 9
<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

\eledsection@correcting@skip	1
\eledsectmark	1, 20
\eledsectnotoc	1, 20
\endlock	1
\endnumbering	1, 15
\endnumberingR	1
\endsub	1
environments:	
astanza	18
Leftside	15
pages	9
pairs	8
Rightside	15

F

\f@x@l@cksR	1
\finish@Pages@notes	1
\first@linenum@out@Rfalse	1
\first@linenum@out@Rtrue	1
\firstlinenum	1, 16
\firstlinenum*	1, 16
\firstlinenumR	1, 16
\firstsublinenum	1, 16
\firstsublinenum*	1, 16
\firstsublinenumR	1, 16
\fix@page	1
\flag@end	1
\flag@start	1
\flush@notesR	1
\footnote@reading	1
\footnote@typeset	1
\footnoteXmk	14
\footnoteXnomk	14

G

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

H

\hidenumbering	16
----------------	----

I

\if@getnextbox	1
----------------	---

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

L

<code>\l@d@set</code>	1
<code>\l@dbfnote</code>	1
<code>\l@dc@maxchunks</code>	1
<code>\l@dcalc@maxoftwo</code>	1
<code>\l@dcalc@minoftwo</code>	1
<code>\l@dcalcnum</code>	1
<code>\l@dchecklang</code>	1
<code>\l@dleftbox</code>	1
<code>\l@dlinenumR</code>	1
<code>\l@dmake@labelsR</code>	1
<code>\l@dminpagelines</code>	1
<code>\l@dnumpstartsL</code>	1
<code>\l@dnumpstartsR</code>	1
<code>\l@dpagfullfalse</code>	1
<code>\l@dpagfulltrue</code>	1
<code>\l@drightbox</code>	1

<code>\l@dsamepagefalse</code>	1
<code>\l@dsamepagetrue</code>	1
<code>\l@dsetupmaxlinecounts</code>	1
<code>\l@dsetuprawboxes</code>	1
<code>\l@dskipversenumberR</code>	1
<code>\l@dusedbabelfalse</code>	1
<code>\l@dusedbabeltrue</code>	1
<code>\l@duselanguage</code>	1
<code>\l@deromaxlinecounts</code>	1
<code>\l@pscL</code>	1
<code>\l@pscR</code>	1
<code>\labelref@listR</code>	1
<code>\last@page@numR</code>	1
<code>\Lcolwidth</code>	1, 8, 10
<code>\led@err@BadLeftRightPstarts</code>	1
<code>\led@err@Columns@InsideEnv</code>	1
<code>\led@err@Columns@WithoutEnv</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@Pages@WithoutEnv</code>	1
<code>\led@err@RightOnLeftPage</code>	1
<code>\led@err@Rightside@PreviousNotPrinted</code>	1
<code>\led@err@TooManyPstarts</code>	1
<code>\led@error@fail@patch@memnum</code>	1
<code>\led@error@fail@patch@outputpage</code>	1
<code>\led@error@fail@patch@pagenumbering</code>	1
<code>\led@error@fail@patch@thepage</code>	1
<code>\led@nopbnumR</code>	1
<code>\led@nopbR</code>	1
<code>\led@pbnumR</code>	1
<code>\led@pbR</code>	1
<code>\led@warn@ChangeSyncOption</code>	1
<code>\led@warn@setting@in@rightside</code>	1
<code>\lednopbnum</code>	1
<code>\lednopbnumR</code>	1
<code>\ledpbnumR</code>	1
<code>\ledpbR</code>	1
<code>\ledstrutL</code>	1
<code>\ledstrutR</code>	1
<code>\ledthegoal</code>	1
<code>\leftlinenumR</code>	1
<code>\leftstartnumL</code>	1
<code>\leftstartnumR</code>	1
<code>Leftside (environment)</code>	15
<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, 16
<code>\lineationR</code>	1, 16
<code>\linenum@outR</code>	1
<code>\linenumberstyle*</code>	1, 16
<code>\linenumberstyleR</code>	1, 16
<code>\linenumincrement</code>	1, 16
<code>\linenumincrement*</code>	1, 16
<code>\linenumincrementR</code>	1, 16
<code>\linenummargin</code>	1
<code>\linenummargin*</code>	1, 16
<code>\linenummarginR</code>	1, 16
<code>\linenumrepR</code>	1
<code>\linesinpar@listL</code>	1
<code>\linesinpar@listR</code>	1
<code>\list@clearing@regR</code>	1
<code>\list@pstartL@pc</code>	1
<code>\list@pstartR@pc</code>	1
<code>\lock@off</code>	1

M

<code>\maxchunks</code>	1, 7
<code>\maxlinesinpar@list</code>	1
<code>\memorydump</code>	15
<code>\memorydumpL</code>	1
<code>\memorydumpR</code>	1

N

<code>\n@num</code>	1
<code>\namebox</code>	1
<code>\new@lineL</code>	1
<code>\new@lineR</code>	1
<code>\newnamebox</code>	1
<code>\newnamecount</code>	1
<code>\newseries@par</code>	1
<code>\normalbfnoteX</code>	1
<code>\notesXwidthliketwocolumns</code>	9
<code>\num@linesR</code>	1
<code>\numberpstartfalse</code>	16
<code>\numberpstarttrue</code>	16
<code>\numpagelinesL</code>	1
<code>\numpagelinesR</code>	1

O

<code>\one@lineR</code>	1
<code>\onlysideX</code>	14
<code>optionadvancedshiftedpstarts</code>	10, 11
<code>optionnomaxlines</code>	10, 11, 22
<code>optionnosyncpstarts</code>	12, 22, 104
<code>optionshiftedpstarts</code>	6, 11, 22

P

<code>\page@action</code>	1
<code>\page@numR</code>	1
<code>\Pages</code>	1, 10
<code>pages</code> (environment)	9
<code>\Pages@mainmatter</code>	1
<code>pairs</code> (environment)	8
<code>\par@lineR</code>	1
<code>\par@patch@pagenumbering</code>	1
<code>\par@patch@thepage</code>	1
<code>\parledgroup@</code>	1
<code>\parledgroup@beforenotes@save</code>	1
<code>\parledgroup@beforenotesL</code>	1
<code>\parledgroup@beforenotesR</code>	1
<code>\parledgroup@correction@notespacing</code>	1
<code>\parledgroup@correction@notespacing@final</code>	1
<code>\parledgroup@correction@notespacing@init</code>	1
<code>\parledgroup@notes@startL</code>	1
<code>\parledgroup@notes@startR</code>	1
<code>\parledgroup@notespacing@correction</code>	1
<code>\parledgroup@notespacing@set@correction</code>	1
<code>\parledgroupseries@</code>	1
<code>\parledgrouptype@</code>	1
<code>\pausenumberingR</code>	1
<code>\pend</code>	17
<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>\pstart</code>	17
<code>\pstartL</code>	1
<code>\pstartR</code>	1

R

<code>\Rcolwidth</code>	1, 8, 10
<code>\read@linelist</code>	1
<code>\reledpar@error</code>	1
<code>\reledpar@warning</code>	1
<code>\restore@pstartL@pc</code>	1
<code>\restore@pstartR@pc</code>	1
<code>\resumenumberingR</code>	1
<code>\rightlinenumR</code>	1

<code>\rightpstartnumL</code>	1
<code>\rightpstartnumR</code>	1
<code>Rightside</code> (environment)	15
<code>\Rightsidehook</code>	1
<code>\Rightsidehookend</code>	1
<code>\Rlineflag</code>	1

S

<code>\save@familiarfootnote@number</code>	1
<code>\save@section@number</code>	1
<code>\section@numR</code>	1
<code>\selectlanguage</code>	1
<code>\set@line</code>	1
<code>\set@line@action</code>	1
<code>\set@sectcountR</code>	1
<code>\setgoalfraction</code>	1, 12
<code>\sethangingsymbol</code>	18
<code>\setline</code>	1
<code>\setlinenum</code>	1
<code>\setnamebox</code>	1
<code>\setnotepositionliketwocolumns@C</code>	1
<code>\setnotepositionliketwocolumns@L</code>	1
<code>\setnotepositionliketwocolumns@R</code>	1
<code>\setpositionliketwocolumns@C</code>	1
<code>\setpositionliketwocolumns@L</code>	1
<code>\setpositionliketwocolumns@R</code>	1
<code>\setRlineflag</code>	16
<code>\setwidthliketwocolumns@C</code>	1
<code>\setwidthliketwocolumns@L</code>	1
<code>\setwidthliketwocolumns@R</code>	1
<code>\sidenote@marginR</code>	1
<code>\sidenotemargin*</code>	1
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 16
<code>\startlock</code>	1
<code>\startsub</code>	1
<code>\sub@action</code>	1
<code>\subline@numR</code>	1
<code>\sublinenumberstyle*</code>	1, 16
<code>\sublinenumberstyleR</code>	1, 16
<code>\sublinenumincrement</code>	1, 16
<code>\sublinenumincrement*</code>	1, 16
<code>\sublinenumincrementR</code>	1, 16
<code>\sublinenumrepR</code>	1

T

<code>\theledlanguageL</code>	1
<code>\theledlanguageR</code>	1
<code>\thepar@page</code>	1
<code>\thepstartL</code>	16

<code>\thepstartR</code>	16
<code>\thestanzaL</code>	<u>1</u> , 18
<code>\thestanzaR</code>	<u>1</u> , 18

U

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

W

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

X

<code>\Xendlineflag</code>	14
<code>\Xlineflag</code>	14
<code>\Xnoteswidthliketwocolumns</code>	9
<code>\Xonlyside</code>	14

Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added section of babel related code	78
Fix babel problems	1
\Columns: Added \l@dchecklang and \l@duselanguage to \Columns	84
\Pages: Added \l@duselanguage to \Pages	93
v0.3.0.	
General: Added \do@lineLhook and \do@lineRhook	58
Added hooks into Leftside environment	49
Reorganize for ledarab	1
\affixline@numR: Changed \affixline@numR to match neweledmac	62
\do@actions@nextR: Used \do@actions@fixedcode in \do@actionsR	61
\do@lineL: Added \do@lineLhook to \do@lineL	56
Simplified \do@lineL by using macros for some common code	56
\do@lineR: Changed \do@lineR similarly to \do@lineL	59
\flag@end: Removed extraneous spaces from \flag@end	43
\ifledRcol: Moved \ifl@dpairing toeledmac	23
\ifpst@rtedR: Moved \ifpst@rtedL toeledmac	26
\l@dlinenumR: Simplified \leftlinenumR and \rightlinenumR by introducing \l@dlinenumR	34
\l@dnumpstartsR: Moved \l@dnumpstartsL toeledmac	80
\ledstrutR: Added \ledstrutL and \ledstrutR	98
\normalbfnoteX: Removed extraneous spaces from \normalbfnoteX	75
\Pages: Added \ledstrutL to \Pages	94
Added \ledstrutR to \Pages	95
\Rightsidehookend: Added \Leftsidehook, \Leftsidehookend, \Rightsidehook and \Rightsidehookend	50
\sublinenumrepR: Added \linenumrepR and \sublinenumrepR	33
v0.3.a.	
General: Minor \linenummargin fix	1
\line@marginR: Do not just set \line@marginR in \linenummargin	31
v0.3.b.	
General: Improved parallel page balancing	1
\Pages: Added \l@dminpagelines calculation for succeeding page pairs	97
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 unequal length.	1

v0.8.0.	
General: Possibility to have a symbol on each hanging of verses, like in the french typography. Redefine the commande <code>\hangingsymbol</code> to define the character.	1
v0.9.0.	
General: Possibility to number <code>\pstart</code>	16
Possibility to number the pstart with the commands <code>\numberpstarttrue</code>	1
<code>\iflledRcol</code> : Moved <code>\iflledRcol</code> and <code>\ifnumberingR</code> to <code>eledmac</code>	23
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>).	56
Lineation can be by pstart (like in <code>eledmac 0.15</code>).	30
New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . .	75
<code>\affixline@numR</code> : Changed <code>\affixline@numR</code> to allow to disable line numbering (like in <code>eledmac 0.15</code>).	62
<code>\Columns</code> : Line numbering by pstart.	85
<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>).	104
Pstart number can be printed in side	106
<code>\inserthangingsymbolR</code> : Prevent the column separator for hanging verse from shifting	75
v0.12.0.	
General: New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions.	75
v1.0.0.	
General: Compatibility with <code>eledmac</code> . Change name to <code>eledpar</code>	1
Debug in lineation by pstart	30
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>).	51
v1.1.1.	
<code>\pstartR</code> : Correct <code>\pstartR</code> bug introduced by 1.1.	51
v1.1.2.	
<code>\affixside@noteR</code> : Remove spurious space between line number and line content . .	74
v1.2.0.	
General: Support for <code>\led<section></code> commands in parallel texts.	1
v1.2.1.	
<code>\set@sectcountR</code> : For the right section, the counter is defined only once.	28

v1.3.0.		
\edtext:	Manage RTL language.	43
v1.3.1.		
\l@dbfnote:	Compatibility of standard footnotes with eledmac when theses footnotes contain any commands.	75
v1.3.2.		
General:	Debug with some classes.	1
v1.3.3.		
General:	Debugging the left notes of the right column.	74
\l@dbfnote:	Spurious space with footnote in right column.	75
v1.3.4.		
General:	Allow use of commands in sidenotes, as introduced by eledmac 1.0.	74
v1.3.5.		
\normalbfnoteX:	Allows one to redefine \thefootnoteX with alph when some packages are loaded.	75
v1.4.0.		
General:	Added \do@insidelineLhook and \do@insidelineRhook	58
v1.4.1.		
General:	Enable the use of stanzaindentsrepetition within astanza environment.	75
\normalbfnoteX:	Fix bug with normal familiar footnotes when mixing RTL and LTR text.	75
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.	75
\pendL:	Spurious spaces in \pendL.	54
\pendR:	Spurious spaces in \pstartR.	55
\pstartR:	Spurious spaces in \pstartL and \pstartR.	51
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.	32
v1.6.0.		
General:	Add tool and documentation for parallel ledgroups	19
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.	26
\l@dlsnote,\l@drsnote and \l@dcnote	defined only one time, in eledmac.	74
Add \beforecolumnseparator and \aftercolumnseparator.	9
Add \columnspostion.	9
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.	56
Compatibility with musixtex.	1
Debug eledmac sectioning command after using \resumenumbering.	1
New sectioning commands, as in eledmac.	20
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.	78
\Columns:	Modify \Columns to enable to add section's title.	83

Suppress \l@dchecklang from \Columns.	84
\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.	78
\Pages: Modify \Pages to enable to add section's title.	91
\pendL: As in eledmac, \pendL can have an optional argument.	54
\pendR: As in eledmac, \pendR can have an optional argument.	55
\print@columnseparator: Move some code of \Columns to \print@columnseparator.	86
\pstartR: As in eledmac, \pendL and \pendR can have an optional argument.	51
\sidenotemargin*: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar.	73
Add \sidenotemargin*	73
\theledlanguageR: Correct left/right language setting with polyglossia.	79
v1.8.1.	
\do@lineL: Fix a bug with critical notes at the beginning of a page, (maybe added by v1.8.0) (?).	56
\do@lineR: Fix a bug with critical notes at the beginning of a page, added by v1.8.0 (?).	59
v1.8.2.	
General: Debug \eledxxx with some paper sizes	1
Debug left and side note (bugs added by 1.8.0)	1
\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar	43
\lineation*: Add \lineation*	30
\reledpar@error: Errors specific to eledpar send to eledpar handbook	24
v1.8.3.	
General: Add \noeledxxx, as in eledmac	1
\doinsidelineRhook: Added \dolineLhook, \dolineRhook, \doinsidelineLhook and \doinsidelineRhook	58
\Pages: Debug blank pages when using optional argument in the last \pend.	91
\resumenumberingR: Debug \resumenumberingR	29
v1.9.0.	
General: Add \AtBeginPairs macro.	8
Compatibility with \Xnoteswidthliketwocolumns and \notesXwidthliketwocolumns	1
\ifwidthliketwocolumns: Added widthliketwocolumns option	23
\theledlanguageR: Debug left/right language switching with polyglossia. Do not write in .aux file when setting left/right lines.	79
v1.9.1.	
\ifledRcol: Moved \ifl@dpaging to eledmac	23
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).	91
Debug wrong parallel pages synchronization when an \edtext falls across two pages.	91
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.	42

v1.11.0.	
General: Compatibility of standard footnotes with some biblatex styles.	1
\edtext: \critext and \edtext are now defined only in eledmac.	43
v1.12.0.	
General: Compatibility with Lua \TeX RTL languages.	1
\Columns: Add \l@dprintingcolumnstrue	83
\edlabel: \edlabel and \edindex works now with hyperref when using eledpar. . .	73
\edlabel is now defined only one time for both eledmac and eledpar	73
\Pages: Add \l@dprintingpagetrue	91
\print@eledsectionL: Compatibility with Lua \TeX RTL languages.	58
\print@eledsectionR: Compatibility with Lua \TeX RTL languages.	59
\print@lineL: Compatibility with Lua \TeX RTL languages.	57
v1.12.1.	
\print@eledsectionL: Fixes bug with Lua \TeX RTL \eledsection.	58
v1.13.0.	
General: Enable the use of optional argument of & in astanza environment.	75
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.	99
\ifledRcol: Remove false boolean settings which are not needed.	23
\Pages: Prevent false overfull hboxes when using \Pages outside of pages environment.	92
When using shiftedpstarts option, a \l@dleftbox with a null height will advance the \pagetotal in any case.	91
v1.13.1.	
\correct@footinsX@box: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages.	68
Correct \correct@footinsX@box and \correct@Xfootins@box	68
\Pages: Prevent false empty page after \Pages (bug added in 1.13.0)	91
v1.14.0.	
General: Fix bug with line number position when using \eledsection and similar com- mands for RTL texts with Lua \TeX	1
The \newifs are not followed by boolean values set to false, because it is the \TeX default setting.	1
v1.15.0.	
General: Add \AtEveryPstartCall.	1
Add sameparallepagenumber option.	12
Fix vertical spurious space before right \eledchapter (bug added in v1.13.0).	1
Prevent vertical space when using \AtEveryPstart or \AtEveryPend with a com- mand which prints nothing	1
\do@actions@nextR: Add action 1008 and 1009	61
\inserthangingsymbolR: Prevent more efficiently the column separator from shifting when a verse is hanging	75
\lineationR: As \lineation, \lineationR automatically set the \pstartinfootnote.	30
\n@num: \n@num defined only one time for both Eledmac and Eledpar.	39
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar	43

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.	44
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	16
v2.0.0.	
\@adv: \@adv defined only in reledmac.	37
\@lab: \@lab defined only in eledmac.	73
\@ref@regR: \@ref defined only in reledmac, code specific to right side moved in \ref@regR.	39
\@set: \@set defined only in reledmac.	37
General: \@nl is now defined only in reledmac.	36
\ifbypage@ and \ifbypstart@R defined in eledmac.	30
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.	43
\bbl@set@language: Patch \bbl@set@language instead of redefining it	78
\do@lockonR: \do@lockon defined only in reledmac.	37
\endlock: \startlock and \endlock defined only in reledmac.	43
\endsub: \startsub and \endsub defined only in reledmac.	43
\fix@page: \fix@page is defined only once in reledmac	37
chapterinpages: Deleting the old system of managing parallel chapter, keep only the new one with \patchcmd.	49
\l@d@set: \l@d@set defined only in reledmac.	37
\l@dbfnote: \l@dbfnote defined only in reledmac.	75
\line@marginR: \linenummargin now defined only once time in reledmac.	31
\normalbfnoteX: \normalbfnoteX defined only in reledmac.	75
\page@action: \page@action defined only in reledmac.	37
\read@linelist: \read@linelist is defined only once time in \reledmac.	36
\set@line: \set@line defined only in reledmac.	43
\set@line@action: \set@line@action defined only in reledmac.	37
\setline: \setline defined only in reledmac.	43
\setlinenum: \setlinenum defined only in reledmac.	43

\skip@lockoff: \do@lockoff defined only in reledmac.	38
\sub@action: \sub@action defined only in reledmac.	37
\sublinenumincrement*: \firstlinenum, \linenumincrement, \firstsublinenum, \sublinenumincrement are now defined only in reledmac.	32
\theledlanguageR: Patch \otherlanguage instead of redefining it.	79
v2.1.0.	
General: Fix bug when using \eledsection and related on right pages when page width is short.	1
Fix bug when using \pagenumbering with memoir (bug added in v2.0.0).	1
Fix bug with \setparledgroupnotespacing with the shiftedpstarts option.	1
Fix incompatibility between optional argument of \pstart and \numberpstarttrue	1
Options to custom empty right page before \Pages.	1
v2.2.0.	
General: astanza environment can take an optional argument, which will be the optional argument of \pstart started by this environment.	1
New tools to number stanza	1
v2.2.1.	
General: Fix bug with optional argument of last left \pend	1
v2.3.0.	
General: Change some internal codes in order to provide compatibility with L ^A T _E X release of october 2015	1
Fix bug with title number in parallel columns	1
New line setting command suffixed by R to set only the right side.	1
\Pages: Fix bug when calling \Columns after a \Pages (bug added in v1.13.0).	92
v2.4.0.	
General: New way of (not) synchronizing the parallel pages.	1
Option to switch to \mainmatter when calling \Pages	1
\ledstrutR: Deleted \ledstrutL and \ledstrutR	98
Fix bug with dotted letter	98
v2.5.0.	
General: Disable empty lines as paragraph in astanza.	1
Fix bug introduced in v1.15.0 which made hanging indentation in verse not work any- more.	1
New commands \linenummarginR and \linenummargin*	1
v2.5.1.	
General: Fix spurious space when using optional argument of astanza environment (in- troduced in v2.5.0).	1
v2.5.2.	
General: Fix bug introduced in v2.5.0 with \linenummargin, \firstlinenum, \linenumincrement, \firstsublinenum, \sublinenumincrement.	1
v2.6.0.	
General: \Xlineflag and \Xendlineflag added	1
\printlinesR deleted	1
Error message when calling \Pages or \Columns without previous pages or pairs environment.	1
Fix bug with footnote numbering when using the same series of familiar footnotes on both sides.	1
Fix bug with right side title number when using title commands before pages or columns environments.	1
Fix compatibility with babel (broken in v2.0.0).	1

No error messages about ends of left / right page when using the <code>\syntonly</code> command of the <code>syntonly</code> package.	1
<code>\l@dmake@labelsR: \@Rlineflag</code> is not stored directly after the line number, but as a fifth argument of <code>\the@labelX</code> . Can be retrieved by <code>\xflagref</code>	73
v2.6.1.	
General: Fix bug, introduced in v2.6.0, with footnote numbering when using <code>perpage</code> package.	1