

# reledmac

## Typeset scholarly editions with L<sup>A</sup>T<sub>E</sub>X\*

Maïeul Rouquette<sup>†</sup>

based on the original ledmac by

Peter Wilson

Herries Press

which was based on the original edmac, tabmac and edstanza by

John Lavagnino, Dominik Wujastyk, Herbert Breger and Wayne Sullivan.

### Abstract

The `reledmac` provides many tools in order to typeset scholarly editions. It is based on the `eledmac` package, which was based on the `ledmac` package, which was based on the `edmac` T<sub>E</sub>X package.

It can be used in combination with `reledpar` in order to typeset two texts in parallel, like an original text and its translation in a modern language.

`reledmac` 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 every possible case). Examples starting with “1-” are for basic uses, those starting with “2-” are for advanced uses.

To report bugs or request a new feature, please go to ledmac GitHub page and click on “New Issue”: <https://github.com/maieul/ledmac/issues/>. You must create an account on github.com to access my page (maieul/ledmac). GitHub accounts are free for open-source users. You can post messages in English or in French (preferred).

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

## Contents

<b>1 Introduction</b>	<b>10</b>
1.1 Aim of the package	10
1.2 History	11
1.2.1 edmac	11
1.2.2 ledmac	13
1.2.3 eledmac	13

---

\*This file (`reledmac.dtx`) has version number v2.9.0, last revised 2016/03/23.

<sup>†</sup>maieul at maieul dot net

1.2.4 <code>reledmac</code> . . . . .	13
1.3 List of works edited with (r)(e)ledmac . . . . .	13
<b>2 How the package works</b>	<b>13</b>
<b>3 Options</b>	<b>14</b>
3.1 Specific features . . . . .	14
3.2 Optimizing package performance . . . . .	15
<b>4 Text lines and paragraphs numbering</b>	<b>15</b>
4.1 Text lines numbering . . . . .	15
4.2 Paragraphs . . . . .	16
4.2.1 Basics . . . . .	16
4.2.2 Automatically producing <code>\pstart ... \pend</code> . . . . .	16
4.2.3 Content before specific <code>\pstart</code> and after specific <code>\pend</code> . . . . .	17
4.2.4 Content before every <code>\pstart</code> and after every <code>\pend</code> . . . . .	17
4.2.5 Numbering paragraphs ( <code>\pstart</code> ) . . . . .	17
4.2.6 Languages written in Right to Left . . . . .	18
4.2.7 Memory limits . . . . .	18
4.3 Lineation commands . . . . .	18
4.3.1 Disabling lineation . . . . .	18
4.3.2 Setting lineation start and step . . . . .	19
4.3.3 Setting lineation reset . . . . .	19
4.3.4 Setting line number margin . . . . .	19
4.3.5 Other settings . . . . .	20
4.4 Changing the line numbers . . . . .	20
4.4.1 Sublineation . . . . .	20
4.4.2 Locking lineation . . . . .	20
4.4.3 Setting and changing line number . . . . .	20
4.4.4 Line number style . . . . .	21
4.4.5 Skipping and hiding number . . . . .	21
4.4.6 Execute code at each line . . . . .	21
<b>5 Apparatus commands</b>	<b>22</b>
5.1 Terminology . . . . .	22
5.2 Critical notes . . . . .	22
5.2.1 The lemma . . . . .	22
5.2.2 Footnotes . . . . .	23
5.2.3 Endnotes . . . . .	23
5.2.4 Paragraph in critical apparatus . . . . .	24
5.2.5 Change lemma and line number . . . . .	24
5.2.6 Changing the names of commands for critical apparatus . . . . .	25
5.3 Disambiguation of identical words in the apparatus . . . . .	26
5.3.1 Basic use . . . . .	26
5.3.2 Notes about input encoding with UTF-8 processor . . . . .	26
5.3.3 Use with <code>\lemma</code> command . . . . .	27

5.3.4 Customizing . . . . .	28
5.4 Familiar notes . . . . .	28
5.4.1 Basic use . . . . .	28
5.4.2 Customizing mark . . . . .	28
5.4.3 Separator for multiple footnotes . . . . .	29
5.5 Changing series . . . . .	29
5.5.1 Create a new series . . . . .	29
5.5.2 Delete series . . . . .	29
5.5.3 Series order . . . . .	29
5.6 Position of critical and familiar footnotes . . . . .	29
<b>6 Critical apparatus appearance</b> . . . . .	<b>30</b>
6.1 Notes arrangement in a series . . . . .	30
6.2 Control line number printing . . . . .	31
6.2.1 Print line number only at first time . . . . .	31
6.2.2 Arbitrary text before line number . . . . .	31
6.2.3 Separator for line range . . . . .	31
6.2.4 Abbreviate line range . . . . .	31
6.2.5 Disable line number . . . . .	32
6.2.6 Printing pstart number . . . . .	32
6.2.7 Printing stanza number . . . . .	33
6.2.8 Separator between line and subline numbers . . . . .	33
6.2.9 Space around number . . . . .	33
6.2.10 Space around line symbol . . . . .	33
6.2.11 Space in place of number . . . . .	34
6.2.12 Boxing line number and line symbol . . . . .	34
6.3 For endnotes . . . . .	35
6.4 Arbitrary code around line number . . . . .	35
6.5 Separator between the lemma and the note . . . . .	35
6.5.1 For footnotes . . . . .	35
6.5.2 For endnotes . . . . .	36
6.6 Font style . . . . .	36
6.6.1 For line number . . . . .	36
6.6.2 For the lemma . . . . .	36
6.6.3 For all notes . . . . .	37
6.7 Indent of notes content . . . . .	37
6.8 Arbitrary code at the beginning of notes . . . . .	37
6.9 Options for footnotes in columns . . . . .	37
6.9.1 Alignment . . . . .	37
6.9.2 Size of the columns . . . . .	38
6.10 Options for paragraphed footnotes . . . . .	38
6.10.1 Mark separation of notes . . . . .	38
6.10.2 Ragged text . . . . .	38
6.11 Options for block of notes . . . . .	39
6.11.1 Text before notes . . . . .	39
6.11.2 Code before notes . . . . .	39

6.11.3 Spacing . . . . .	39
6.11.4 Rule . . . . .	39
6.11.5 Maximum height . . . . .	39
6.11.6 Width . . . . .	40
6.12 Footnotes and the <code>reledpar</code> columns . . . . .	40
6.13 Endnotes in one paragraph . . . . .	40
<b>7 Fonts</b>	<b>41</b>
<b>8 Verse</b>	<b>41</b>
8.1 Basic . . . . .	41
8.2 Define stanza indents . . . . .	41
8.3 Repeating stanza indents . . . . .	42
8.4 Manual stanza indent . . . . .	43
8.5 Stanza breaking . . . . .	43
8.6 Hanging symbol . . . . .	43
8.7 Long verse and page break . . . . .	44
8.8 Content before/after verses . . . . .	44
8.9 Numbering stanza . . . . .	44
8.10 Various tools . . . . .	44
8.11 Notes on empty lines . . . . .	45
<b>9 Grouping</b>	<b>45</b>
<b>10 Cross referencing</b>	<b>45</b>
10.1 Basic use . . . . .	45
10.2 Cross-referencing to a critical note . . . . .	46
10.3 Cross-referencing which return a number in any case . . . . .	46
10.3.1 Cross-referencing in order to define line number of a critical note . . . . .	47
10.4 Not automatic cross-referencing . . . . .	47
10.5 Normal $\LaTeX$ cross-referencing . . . . .	47
10.6 References to start and end lines . . . . .	47
10.6.1 Reference to main text lines . . . . .	47
10.6.2 References to lines that are commented on in the apparatus . . . . .	48
10.6.3 Settings . . . . .	48
10.7 Compatibility with <code>xr</code> package . . . . .	50
<b>11 Side notes</b>	<b>50</b>
11.1 Basics . . . . .	50
11.2 Setting . . . . .	50
11.2.1 Width . . . . .	50
11.2.2 Vertical position . . . . .	50
11.2.3 Distance to the main text . . . . .	51
11.2.4 Separator between notes . . . . .	51

<i>Contents</i>	5
<b>12 Indexing</b>	<b>51</b>
12.1 Basics	51
12.2 Referring to critical notes	51
12.3 Separator between page and line numbers	52
12.4 Using xindy	52
12.5 Advanced setting	53
<b>13 Glossary</b>	<b>53</b>
13.1 Preamble setting	53
13.2 Commands	53
<b>14 Tabular material</b>	<b>54</b>
<b>15 Sectioning commands</b>	<b>57</b>
15.1 Sectioning commands without line numbers or critical notes	57
15.2 Sectioning commands with line numbering and critical notes	57
15.3 Optimization	58
<b>16 Quotation environments</b>	<b>58</b>
<b>17 Page breaks</b>	<b>58</b>
17.1 Control page breaking	58
17.2 Prevent page break in a long verses	59
<b>18 Miscellaneous</b>	<b>59</b>
18.1 Known and suspected limitations	60
18.1.1 floatrow package compatibility	60
18.1.2 ‘No room for a new’	60
18.1.3 Marginal notes	60
18.1.4 Paragraph shape	60
18.1.5 Paragraphed footnotes	61
18.1.6 Use with other packages	61
18.1.7 Parallel typesetting	62
<b>I Implementation overview</b>	<b>63</b>
<b>II Preliminaries</b>	<b>63</b>
II.1 Links with original edmac	63
II.2 Package declaration	63
II.3 Package options	64
II.4 Loading packages	65
II.5 Compatibility with Lua $\TeX$	66
II.6 Boolean flags	66
II.7 Messages	67
II.8 Gobbling	72
II.9 Miscellaneous commands	73
II.10 Prepare reledpar	73

II.11 Booleans provided by other optional packages which are required in any case . . . . .	74
<b>III Sectioning commands</b>	<b>74</b>
<b>IV List macros</b>	<b>78</b>
<b>V Line counting</b>	<b>79</b>
V.1 Choosing the system of lineation . . . . .	79
V.2 Line number margin . . . . .	81
V.3 Line number initialization and increment . . . . .	82
V.4 Line number locking . . . . .	83
V.5 Line number style . . . . .	84
V.6 Line number printing . . . . .	85
V.7 Line number counters and lists . . . . .	86
V.8 Line number locking counter . . . . .	87
V.9 Line number associated to lemma . . . . .	87
V.10 Reading the line-list file . . . . .	90
V.11 Commands within the line-list file . . . . .	92
V.12 Writing to the line-list file . . . . .	103
<b>VI Marking text for notes</b>	<b>108</b>
VI.1 <code>\edtext</code> itself . . . . .	109
VI.2 Substitute lemma . . . . .	116
VI.3 Substitute line numbers . . . . .	117
VI.4 Lemma disambiguation . . . . .	118
<b>VII Paragraph decomposition and reassembly</b>	<b>123</b>
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	124
VII.2 Processing one line . . . . .	129
VII.2.1 General process . . . . .	129
VII.2.2 Process for “normal” line . . . . .	129
VII.2.3 Process for line containing <code>\eledsection</code> command . . . . .	131
VII.2.4 Hooks . . . . .	131
VII.2.5 Sidenotes and marginal line number initialization . . . . .	131
<b>VIII Line and page number computation</b>	<b>132</b>
<b>IX Line number printing</b>	<b>135</b>
<b>X Pstart number printing in side</b>	<b>139</b>
<b>XI Restoring footnotes and penalties</b>	<b>140</b>
XI.1 Add insertions to the vertical list . . . . .	141
XI.2 Penalties . . . . .	142
XI.3 Printing leftover notes . . . . .	142

<b>XII Critical footnotes</b>	<b>143</b>
XII.1 Fonts . . . . .	143
XII.2 Individual note options . . . . .	144
XII.3 Notes language . . . . .	144
XII.4 General survey of the way we manage notes . . . . .	145
XII.5 General setup . . . . .	146
XII.6 Footnotes arrangement . . . . .	147
XII.6.1 User level macro . . . . .	147
XII.6.2 Normal footnote . . . . .	147
XII.6.3 Paragraphed footnotes . . . . .	151
XII.6.4 Columnar footnotes . . . . .	158
XII.7 Critical notes presentation . . . . .	165
XII.7.1 Font tools . . . . .	165
XII.7.2 Pstart number in footnote . . . . .	166
XII.7.3 Line number printing . . . . .	166
<b>XIII Familiar footnotes</b>	<b>175</b>
XIII.1 Adjacent footnotes . . . . .	175
XIII.2 Regular footnotes for numbered texts . . . . .	176
XIII.3 Footnote formats . . . . .	178
XIII.4 Footnote arrangement . . . . .	179
XIII.4.1 User level macro . . . . .	179
XIII.4.2 Normal footnotes . . . . .	179
XIII.4.3 Two columns footnotes . . . . .	185
XIII.4.4 Three columns footnotes . . . . .	187
XIII.4.5 Paragraphed footnotes . . . . .	189
XIII.5 Wrapping footnote marks in hyperlink . . . . .	193
<b>XIV Code common to both critical and familiar footnote in normal arrangement</b>	<b>194</b>
<b>XV Footnotes' width for two columns</b>	<b>194</b>
<b>XVI Footnotes' order</b>	<b>196</b>
<b>XVII Footnotes' rule</b>	<b>196</b>
<b>XVIII Specific skip for first series of footnotes</b>	<b>196</b>
XVIII.0.1 Overview . . . . .	196
XVIII.0.2 User level command . . . . .	197
XVIII.0.3 Internal commands . . . . .	198
<b>XIX Endnotes</b>	<b>199</b>

<b>XX Generate series of notes</b>	<b>207</b>
XX.1 Test if series is still existing . . . . .	207
XX.2 Init specific to <code>reledpar</code> . . . . .	207
XX.3 For critical footnotes . . . . .	208
XX.3.1 Options . . . . .	208
XX.3.2 Create inserts, needed to add notes in foot . . . . .	209
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc. . . . .	209
XX.3.4 Set standard display . . . . .	212
XX.4 For familiar footnotes . . . . .	212
XX.4.1 Options . . . . .	212
XX.4.2 Create tools for familiar footnotes ( <code>\footnotex</code> ) . . . . .	212
XX.5 The endnotes . . . . .	214
XX.5.1 The auxiliary file . . . . .	214
XX.5.2 The main macro . . . . .	214
XX.5.3 The options . . . . .	215
XX.6 Init standards series (A,B,C,D,E) . . . . .	217
<b>XXI Setting series display</b>	<b>217</b>
XXI.1 Change series order . . . . .	217
XXI.2 Test series order . . . . .	217
XXI.2.1 Get the first series . . . . .	217
XXI.3 Series setting . . . . .	218
XXI.3.1 General way of working . . . . .	218
XXI.3.2 Tools to set options . . . . .	218
XXI.3.3 Tools to generate options commands . . . . .	220
XXI.3.4 Options for critical notes . . . . .	221
XXI.3.5 Options for familiar notes . . . . .	223
XXI.3.6 Options for endnotes . . . . .	223
XXI.4 Hooks for a particular footnote . . . . .	225
XXI.5 Alias . . . . .	226
<b>XXII Output routine</b>	<b>226</b>
XXII.0.1 Page number management . . . . .	226
XXII.0.2 Extra footnotes output . . . . .	226
XXII.0.3 Standard output's commands patching . . . . .	229
<b>XXIII Cross referencing</b>	<b>231</b>
XXIII.1 Compatibility with <code>xref</code> . . . . .	244
<b>XXIV Side notes</b>	<b>245</b>
<b>XXV Minipages and such</b>	<b>251</b>



<b>XXVI Indexing</b>	<b>256</b>
XXVI.1 Looking on package order . . . . .	256
XXVI.2 Auxiliary macros for <code>\edindex</code> . . . . .	256
XXVI.3 Code specific to <code>\edindexin</code> critical footnotes . . . . .	257
XXVI.4 Analysis of command in indexed text . . . . .	258
XXVI.5 Code for the formatted index . . . . .	259
XXVI.6 Main code . . . . .	259
XXVI.7 Hyperlink . . . . .	260
XXVI.8 ‘innote’ and ‘notenumber’ option of <code>indextols</code> package . . . . .	263
<b>XXVII Glossaries</b>	<b>264</b>
<b>XXVIII Verse</b>	<b>265</b>
XXVIII.1 Hanging symbol management . . . . .	265
XXVIII.2 Using <code>&amp;</code> character . . . . .	266
XXVIII.3 Code category setting . . . . .	266
XXVIII.4 Stanza count and indent . . . . .	267
XXVIII.5 Numbering stanza . . . . .	268
XXVIII.6 Stanza number in note . . . . .	269
XXVIII.7 Main work . . . . .	270
XXVIII.8 Restore catcode and penalties . . . . .	271
<b>XXIX Arrays and tables</b>	<b>272</b>
XXIX.1 Preamble: macro as environment . . . . .	272
XXIX.2 Tabular environments . . . . .	275
XXIX.2.1 Disabling and restoring commands . . . . .	275
XXIX.2.2 Counters, boxes and lengths . . . . .	279
XXIX.2.3 Tabular typesetting . . . . .	282
XXIX.2.4 Environments . . . . .	294
<b>XXX Quotation’s commands</b>	<b>294</b>
<b>XXXI Section’s title commands</b>	<b>295</b>
XXXI.1 Commands to disable some feature . . . . .	295
XXXI.2 General overview . . . . .	295
XXXI.3 <code>\beforeeledchapter</code> command . . . . .	296
XXXI.4 Auxiliary commands . . . . .	297
XXXI.5 Patching standard commands . . . . .	297
XXXI.6 Main code of <code>\eledxxx</code> commands . . . . .	302
XXXI.7 Macros written in the auxiliary file . . . . .	304
<b>XXXII Page breaking or no page breaking depending of specific lines</b>	<b>307</b>
<b>XXXIII Long verse: prevents being separated by a page break</b>	<b>308</b>
<b>XXXIV Tools for hyperref package</b>	<b>309</b>

<b>XXXV Compatibility with eledmac</b>	<b>310</b>
<b>Appendix A Things to do when changing versions</b>	<b>312</b>
Appendix A.1 Migrating from edmac to ledmac . . . . .	312
Appendix A.2 Migration from ledmac to eledmac . . . . .	313
Appendix A.3 Migration to eledmac 1.5.1 . . . . .	314
Appendix A.4 Migration to eledmac 1.12.0 . . . . .	314
Appendix A.5 Migration to eledmac 17.1 . . . . .	315
Appendix A.6 Migration to eledmac 1.21.0 . . . . .	315
Appendix A.6.1 <code>\Xledsetnormalparstuff</code> and <code>\ledsetnormalparstuff</code>	315
Appendix A.6.2 Endnotes . . . . .	315
Appendix A.7 Migration to eledmac 1.22.0 . . . . .	315
Appendix A.8 Migration to eledmac 1.23.0 . . . . .	315
Appendix A.9 Migration from eledmac to reledmac . . . . .	316
Appendix A.9.1 Risk of ‘no room for a new’ . . . . .	316
Appendix A.9.2 Multiple indices with memoir . . . . .	316
Appendix A.9.3 Deprecated commands and options . . . . .	316
Appendix A.9.4 <code>\renewcommandreplaced</code> by <code>command</code> . . . . .	317
Appendix A.9.5 Commands the names of which have been changed . . . . .	317
Appendix A.9.6 Endnotes . . . . .	319
Appendix A.9.7 Z Series . . . . .	319
Appendix A.9.8 Internal commands . . . . .	319
Appendix A.10 Migration to reledmac 2.1.0 . . . . .	319
Appendix A.11 Migration to reledmac 2.1.3 . . . . .	319
Appendix A.12 Migration to reledmac 2.3.0 . . . . .	319
Appendix A.13 Migration to reledmac 2.4.0 . . . . .	320
Appendix A.14 Migration to reledmac 2.5.0 . . . . .	320
Appendix A.15 Migration to reledmac 2.7.0 . . . . .	320
Appendix A.16 Migration to reledmac 2.7.2 . . . . .	320
Appendix A.17 Migration to reledmac 2.8.0 . . . . .	320
<b>References</b>	<b>321</b>
<b>Index</b>	<b>321</b>
<b>Change History</b>	<b>365</b>

## 1 Introduction

### 1.1 Aim of the package

The `reledmac` package, together with `LATEX`, provides several important facilities for formatting critical editions of texts in a traditional manner. Major features include:

- automatic stepped line numbering, by page, section or paragraph;
- sub-lineation within the main series of line numbers;

- variant readings automatically keyed to line numbers;
- caters to both prose and verse;
- multiple series of footnotes and endnotes;
- block or columnar formatting of the footnotes;
- simple tabular material may be line numbered;
- indexing keyed to page and line numbers.

`reledmac` allows the scholar engaged in preparing a critical edition to focus attention wholly on the task of creating the critical text and evaluating the variant readings, text-critical notes and testimonia.  $\LaTeX$  and `Eledmac` will take care of the formatting and visual correlation of all the disparate types of information.

Apart from `reledmac` there are other  $\LaTeX$  packages for typesetting critical editions. However, the aim of `reledmac` is to provide an “all in one” and flexible tool in the field of critical editions.

Any suggestions for new features are welcome.

This manual contains a general description of how to use `reledmac` followed by the complete source code and its extensive documentation (in sections I and following, enumerated with Roman numerals). It ends with a list of actions to do when migrating from one version to other, a change history and an index to the source code.

You do not need to read the source code for this package in order to use it; we provide this code primarily for reference, and many of our comments on it repeat material that is also found in earlier sections. But no documentation, however thorough, can cover every question that comes up and many can be answered quickly by consulting the code. On a first reading, we suggest that you read only the general documentation in sections 2, unless you are particularly interested in the innards of `reledmac`.

## 1.2 History

### 1.2.1 `edmac`

The original version of `edmac` was `TEXTED.TEX`, written by John Lavagnino in late 1987 and early 1988 for formatting critical editions of English plays.

John passed these macros on to Dominik Wujastyk who, in September–October 1988, added the footnote paragraphing mechanism, margin swapping and other changes to suit his own purposes, making the style more like that traditionally used for classical texts in Latin and Greek (e.g., the Oxford Classical Texts series). He also wrote some extra documentation and sent the files out to several people. This version of the macros was the first to be called `edmac`.

The present version was developed in the summer of 1990, with the intent of adding necessary features, streamlining and documenting the code, and further generalizing it to make it easily adaptable to the needs of editors in different disciplines. John did most of the general reworking and documentation, with the financial assistance of the Division of the Humanities and Social Sciences, California Institute of Technology. Dominik

adapted the code to the conventions of Frank Mittelbach's `doc` option, and added some documentation, multiple-column footnotes, cross-references, and crop marks.<sup>1</sup> A description by John and Dominik of this version of `edmac` was published as 'An overview of `edmac`: a PLAIN  $\TeX$  format for critical editions', *TUGboat* 11 (1990), pp. 623–643.

From 1991 through 1994, the macros continued to evolve, and were tested at a number of sites. We are very grateful to all the members of the (now defunct) `edmac@mailbase.ac.uk` discussion group who helped us with smoothing out the bugs and infelicities in the macros. Ron Whitney and our anonymous reviewer at the TUG were both of great help in ironing out last-minute wrinkles, while Ron made some important suggestions which may help to make future versions of `edmac` even more efficient. Wayne Sullivan, in particular, provided several important fixes and contributions, including adapting the Mittelbach/Schöpf 'New Font Selection Scheme' for use with PLAIN  $\TeX$  and `edmac`. Another project Wayne has worked on is a DVI post-processor which works with an `edmac` that has been slightly modified to output `\specials`. This combination enables you to recover to some extent the text of each line as ASCII code, facilitating the creation of concordances, an *index verborum*, etc.

As of 1994, we were pleased to be able to say that `edmac` was being used for the real-life book production of several interesting editions, such as the Latin texts of Euclid's *Elements*,<sup>2</sup> an edition of the letters of Nicolaus Copernicus,<sup>3</sup> Simon Bredon's *Arithmetica*,<sup>4</sup> a Latin translation by Plato of Tivoli of an Arabic astrolabe text,<sup>5</sup> a Latin translation of part II of the Arabic *Algebra* by Abū Kāmil Shujā' b. Aslam,<sup>6</sup> the Latin *Rithmarchia* of Werinher von Tegernsee,<sup>7</sup> a middle-Dutch romance epic on the Crusades,<sup>8</sup> a seventeenth-century Hungarian politico-philosophical tract,<sup>9</sup> an anonymous Latin compilation from Hungary entitled *Sermones Compilati in Studio Generali Quinqeeclesiensi in Regno Ungarie*,<sup>10</sup> the collected letters and papers of Leibniz,<sup>11</sup> Theodosius's *Spherics*, the German *Algorismus* of Sacrobosco, the Sanskrit text of the *Kāśikāvṛtti* of Vāmana and Jayāditya,<sup>12</sup> and the English texts of Thomas Middleton's collected works.

<sup>1</sup>This version of the macros was used to format the Sanskrit text in volume I of *Metarules of Pāṇinian Grammar* by Dominik Wujastyk (Groningen: Forsten, 1993).

<sup>2</sup>Gerhard Brey used `edmac` in the production of Hubert L. L. Busard and Menso Folkerts, *Robert of Chester's (?) Redaction of Euclid's Elements, the so-called Adelard II Version*, 2 vols., (Basel, Boston, Berlin: Birkhäuser, 1992).

<sup>3</sup>Being prepared at the German Copernicus Research Institute, Munich.

<sup>4</sup>Being prepared by Menso Folkerts *et al.*, at the Institut für Geschichte der Naturwissenschaften in Munich.

<sup>5</sup>Richard Lorch, Gerhard Brey *et al.*, at the same Institute.

<sup>6</sup>Richard Lorch, 'Abū Kāmil on the Pentagon and Decagon' in *Vestigia Mathematica*, ed. M. Folkerts and J. P. Hogendijk (Amsterdam, Atlanta: Rodopi, 1993).

<sup>7</sup>Menso Folkerts, 'Die *Rithmarchia* des Werinher von Tegernsee', *ibid.*

<sup>8</sup>Geert H. M. Claassens, *De Middelnederlandse Kruisvaartromans*, (Amsterdam: Schiphower en Brinkman, 1993).

<sup>9</sup>Emil Hargittay, *Csáky István: Politica philosophiai Okoskodás-szerint való rendes életnek példája (1664–1674)* (Budapest: Argumentum Kiadó, 1992).

<sup>10</sup>Being produced, as was the previous book, by Gyula Mayer in Budapest.

<sup>11</sup>Leibniz, *Sämtliche Schriften und Briefe*, series I, III, VII, being edited by Dr. H. Breger, Dr. N. Gädeke and others at the Leibniz-Archiv, Niedersächsische Landesbibliothek, Hannover. (see <http://www.nlb-hannover.de/Leibniz>)

<sup>12</sup>Being prepared at Poona and Lausanne Universities.

### 1.2.2 ledmac

Version 1.0 of `tabmac` was released by Herbert Breger in October 1996. This added the capability for typesetting tabular material.

Version 0.01 of `edstanza` was released by Wayne Sullivan in June 1992, to help a colleague with typesetting Irish verse.

In March 2003 Peter Wilson started an attempt to port `edmac` from TeX to LaTeX. The starting point was `edmac` version 3.16 as documented on 19 July 1994 (available from CTAN). In August 2003 the `tabmac` functions were added; the starting point for these being version 1.0 of October 1996. The `edstanza` (v0.01) functions were added in February 2004. Sidenotes and regular footnotes in numbered text were added in April 2004. This port was called `ledmac` (L<sup>A</sup>T<sub>E</sub>X `edmac`).

Since July 2011, `ledmac` is maintained by Maïeul Rouquette. It is increasingly powerful and flexible, but it also has become increasingly divergent from the original TeX macro.

### 1.2.3 eledmac

Important changes were put in version 1.0, to make `ledmac` more easily extensible (see 6 p. 30). These changes can trigger small problems with the old customization. That is why a new name was selected: `eledmac` (extended `ledmac`).

To migrate from `ledmac` to `eledmac`, please read Appendix A.2 p. 313.

### 1.2.4 reledmac

`eledmac` has facilitated the creation of customized critical editions. However, the changes made to allow such customization were made in a non-systematic way. Many deprecated commands were kept and many technical ‘debts’ were accumulated, hindering the future evolution of the package.

For these reasons, Maïeul Rouquette decided on a spring cleaning of the code. As some commands name were changed, the resulting compatibility was broken (a little).

A new name was selected: `reledmac` (extended renewed `eledmac`). To migrate from `eledmac` to `reledmac`, please read Appendix A.9 p. 316.

## 1.3 List of works edited with (r)(e)ledmac

A collaborative list of works edited with (r)(e)ledmac is available at [https://www.zotero.org/groups/critical\\_editions\\_typeset\\_with\\_edmac\\_ledmac\\_and\\_eledmac/items](https://www.zotero.org/groups/critical_editions_typeset_with_edmac_ledmac_and_eledmac/items). Please add your own edition made with (r)(e)ledmac.

## 2 How the package works

The `reledmac` package is a three-pass package like L<sup>A</sup>T<sub>E</sub>X itself. Although your textual apparatus and line numbers will be printed on the first run, it takes two more passes through L<sup>A</sup>T<sub>E</sub>X to be sure that everything is correctly placed. If you make any subsequent changes altering the number of lines or notes, the input file may similarly require three

passes to get everything to the right place. `reledmac` will tell you that you need to make more runs when it detects changes, but it does not expend the labor to check this thoroughly. If you have problems with a line or two misnumbered at the top of a page, try running  $\LaTeX$  once or twice more.

However, the best way to be sure to have made the good number of runs is to use some  $\LaTeX$ 's run scripts like *latexmk*.

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.

### 3 Options

The package can be loaded with a number of global options which are listed here. There are two types of options: 1) options which provide specific features, and, 2) options which optimize the package's performance. It is advisable for you to read the relevant parts of the handbook, before reading about the first type of option (specific features), but you can look at the second type (package optimization) in your first reading of the manual.

#### 3.1 Specific features

**draft** underlines lemmas in the main text.

**eledmac-compatible** help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 317).

**nopbinverse** prevents page breaks inside verses.

**noquotation** by default, the quotation environment is redefined inside numbered text. You can disable this redefinition with `noquotation` (see 16 p. 58).

**parapparatus** by default, the apparatus cannot contain paragraph breaks; this option enables paragraphing inside the apparatus.

**xindy** and `xindy+hyperref` are for selecting `xindy` as the index processor (12.4 p. 52).

**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.

## 3.2 Optimizing package performance

**nocritical** disables tools for critical footnotes (`\Afootnote`, `\Bfootnote` etc.). If you do not need critical footnotes, this option lets `eledmac` run faster. It will also preserve room for other packages.

**noeledsec** disables tools for `\eledsection` and related commands (15.2 p. 57).

**noend** disables tools for endnotes (`\Aendnote`, `\Bendnote` etc.). If you do not need endnotes, this option lets `reledmac` run faster. It will also preserve room for other packages.

**nofamiliar** disables tools for familiar footnotes (`\footnoteA`, `\footnoteB` etc.). If you do not need familiar footnotes, this option lets `eledmac` run faster. It will also preserve room for other packages.

**noledgroup** `reledmac` allows use of a series of critical notes and a new series of normal notes inside `minipage` and `ledgroup` environments (see 9 p. 45). However, such features use up computer memory, at the expense of other processing needs. So if you do not need this feature, use `noledgroup` option. This should make `reledmac` faster.

**series** `reledmac` defines five levels of notes: A, B, C, D, E. Using all these levels consumes memory space and processing speed. This is why, if your work does not require the entire A–E series, you can narrow down the available number of series. For example, if you only need A and B series, call the package with `series={A,B}` option.

## 4 Text lines and paragraphs numbering

### 4.1 Text lines numbering

`\beginnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, as in the following example.

```
\beginnumbering
Text
\endnumbering
```

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. The first instance of `\beginnumbering` also opens a file called `<jobname>.<series>end` to receive the text of the endnotes. `\endnumbering` closes the `<jobname>.nn` file.

If the line numbering of a text is to be continuous from start to end, then the whole text will be typed between one pair of `\beginnumbering` and `\endnumbering` commands. But your text will most often contain chapter or other divisions marking sections

that should be independently numbered, and these will be appropriate places to begin new numbered sections.

`reledmac` has to read and store in memory a certain amount of information about the entire section when it encounters a `\beginnumbering` command, so it speeds up the processing and reduces memory use when a text is divided into a larger number of sections (at the expense of multiplying the number of external files that are generated).

## 4.2 Paragraphs

### 4.2.1 Basics

`\pstart` Within a numbered section, each paragraph of numbered text must be marked using the `\pend` `\pstart` and `\pend` commands like this:

```
\pstart
Paragraph of text.
\pend
```

Text that appears within a numbered section but is not marked with `\pstart` and `\pend` will not be numbered.

The following example shows the proper section and paragraph markup and the kind of output that would typically be generated:

```
\beginnumbering
\pstart
This is a sample paragraph, with
lines numbered automatically.
\pend

\pstart
This paragraph too has its
lines automatically numbered.
\pend

The lines of this paragraph are
not numbered.

\pstart
And here the numbering begins
again.
\pend
\endnumbering
```

### 4.2.2 Automatically producing `\pstart ... \pend`

`\autopar` You can use `\autopar` to avoid the nuisance of this paragraph markup and still have every paragraph automatically numbered. The scope of the `\autopar` command needs to be limited by keeping it within a group, as follows:



```

\begingroup
  \beginnumbering
  \autopar

  A paragraph of numbered text.

  Another paragraph of numbered
  text.

  \endnumbering
\endgroup

```

`\autopar` fails, however, on paragraphs that start with a `{` or with any other command that starts a new group before it generates any text. Such paragraphs need to be started explicitly, before the new group is opened, using `\indent`, `\noindent`, or `\leavevmode`, or using `\pstart` itself.<sup>13</sup>

#### 4.2.3 Content before specific `\pstart` and after specific `\pend`

Both `\pstart` and `\pend` can take a optional argument in brackets. Its content will be printed before the beginning of `\pstart` / after the end of `\pend` instead of the argument of `\AtEveryPstart` / `\AtEveryPend`. If you need to start a `\pstart` with brackets, or to add brackets after a `\pend`, just add a `\relax` between `\pstart ... \pend` and the brackets.

This feature is also useful when typesetting verses (see 8 p. 41) or `reledpar` (see 18.1.7 p. 62).

A `\noindent` is automatically added before this argument.

#### 4.2.4 Content before every `\pstart` and after every `\pend`

`\AtEveryPstart`    You can use both `\AtEveryPstart` and `\AtEveryPend`. Their arguments will be  
`\AtEveryPend`    printed before every `\pstart` begins / after every `\pend` ends.

#### 4.2.5 Numbering paragraphs (`\pstart`)

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`.

`\numberpstarttrue`    You can redefine the command `\thepstart` to change style. You can change the value  
`\numberpstartfalse`    of the `pstart` number by using *after* `\beginnumbering`:  
`\thepstart`            `\setcounter{pstart}{value}`

On each `\beginnumbering` the numbering restarts.

`\sidepstartnumtrue`    With the `\sidepstartnumtrue` command, the number of `\pstart` will be printed  
inside. In this case, the line number will be not printed.

`\labelpstarttrue`    With the `\labelpstarttrue` command, a `\label` added just after a `\pstart` will  
refer to the number of this `pstart`.

<sup>13</sup>For a detailed study of the reasons for this restriction, see Barbara Beeton, 'Initiation rites', *TUGboat* 12 (1991), pp. 257–258.

### 4.2.6 Languages written in Right to Left

If you use languages written right to left with Lua $\TeX$  or Xe $\TeX$ , you must switch text direction *before* the `\pstart` command.

### 4.2.7 Memory limits

**This paragraph is kept for history, but the problems described below should not appear with the most recent version of  $\TeX$ .**

`\pausenumbering`  
`\resumenumbering`

reledmac stores a lot of information about line numbers and footnotes in memory as it goes through a numbered section. But at the end of such a section, it empties its memory out, so to speak. If your text has a very long numbered section it is possible that your  $\TeX$  may reach its memory limit. There are two solutions to this.

The first solution is to get a larger  $\TeX$  with increased memory.

The second solution is to split your long section into several smaller ones. The trouble with this is that your line numbering will start again at zero with each new section. To avoid this problem, we provide `\pausenumbering` and `\resumenumbering` which are just like `\endnumbering ... \beginnumbering`, except that they arrange for your line numbering to continue across the break. Use `\pausenumbering` only between numbered paragraphs:

```
\beginnumbering
\pstart
Paragraph of text.
\pend
\pausenumbering

\resumenumbering
\pstart
Another paragraph.
\pend
\endnumbering
```

We have defined these commands as two macros, in case you find it necessary to insert text between numbered sections without disturbing the line numbering. But if you are really just using these macros to save memory, you might as well type,

```
\newcommand{\memorybreak}{\pausenumbering\resumenumbering}
```

and type `\memorybreak` between the relevant `\pend` and `\pstart`.

## 4.3 Lineation commands

### 4.3.1 Disabling lineation

`\numberlinefalse`  
`\numberlinetrue`

Line numbering can be disabled with `\numberlinefalse`. It can be enabled again with `\numberlinetrue`.

### 4.3.2 Setting lineation start and step

`\firstlinenum`  
`\linenumincrement`

By default, `reledmac` numbers every 5th line. There are two counters that control this behaviour: `firstlinenum` and `linenumincrement`. They can be changed using `\firstlinenum{<num>}` and `\linenumincrement{<num>}`. `\firstlinenum` specifies the first line that will have a printed number, and `\linenumincrement` is the difference between successive numbered lines. For example, to start printing numbers at the first line and to have every other line numbered:

```
\firstlinenum{1} \linenumincrement{2}
```

`\firstsublinenum`  
`\sublinenumincrement`  
`\linenumberlist`

There are similar commands, `\firstsublinenum{<num>}` and `\sublinenumincrement{<num>}` for controlling sub-line numbering.

You can define `\linenumberlist` to specify a non-uniform distribution of printed line numbers. For example:

```
\def\linenumberlist{1,2,3,5,7,11,13,17,19,23,29}
```

to have numbers printed on prime-numbered lines only. There must be no spaces within the definition which consists of comma-separated integer numbers. The numbers can be in any order but it is easier to read if you put them in numerical order. Either omitting the definition of `\linenumberlist` or following the empty definition

```
\def\linenumberlist{}
```

the standard numbering sequence is applied. The standard sequence is that specified by the combination of the `firstlinenum`, `linenumincrement`, `firstsublinenum` and `linenumincrement` counter values.

### 4.3.3 Setting lineation reset

`\lineation`

Lines can be numbered either by page, by `pstart` or by section; you specify this using the `\lineation{<arg>}` macro, where `<arg>` is either `page`, `pstart` or `section`.

You may only use this command at places where numbering is not in effect; you can't change the lineation system within a section. You can change it between sections: they don't all have to use the same lineation system. The package's standard setting is `\lineation{section}`. If the lineation is by `pstart`, the `pstart` number will be printed before the line number in the notes.

### 4.3.4 Setting line number margin

`\linenummargin`

The command `\linenummargin{<location>}` specifies the margin where the line (or `pstart`) numbers will be printed. The permissible values for `<location>` are `left`, `right`, `inner`, or `outer`: for example, `\linenummargin{inner}`. The package's default setting is

```
\linenummargin{left}
```

to typeset the numbers in the left hand margin. You can change this whenever you're not in the middle of making a paragraph.

More precisely, the value of `\linenummargin` used is the value in effect at the `\pend` of a numbered paragraph. Apart from an initial setting for `\linenummargin`, only change `\linenummargin` after a `\pend`, whereupon it will apply to all following numbered paragraphs, until changed again (changing it between a `\pstart` and `\pend` pair will apply the change to all of the current paragraph).

### 4.3.5 Other settings

`\leftlinenum` When a marginal line number is to be printed, there are many ways to display it. You can  
`\rightlinenum` redefine `\leftlinenum` and `\rightlinenum` to change the way marginal line numbers  
`\linenumsep` are printed in the left and right margins respectively; the initial versions print the number in font `\numlabfont` (described below) at a distance `\linenumsep` (initially set to one pica) from the text.

## 4.4 Changing the line numbers

Normally, line numbering starts at 1 for the first line of a section and increments by one for each line thereafter. There are various common modifications of this system and the commands described here allow you to put such modifications into effect.

### 4.4.1 Sublineation

`\startsub` You insert the `\startsub` and `\endsub` commands in your text to turn sub-lineation  
`\endsub` on and off. For example, stage directions in plays are often numbered with sub-line numbers: as line 10.1, 10.2, 10.3, rather than as 11, 12, and 13. Titles and headings are sometimes numbered with sub-line numbers as well.

When sub-lineation is in effect, the line number counter is frozen and the sub-line counter advances instead. If one of these commands appears in the middle of a line, it doesn't take effect until the next line; in other words, a line is counted as a line or sub-line depending on what it started out as, even if it changes in the middle.

You can change the separator between line number and subline number or using `\Xsublinesep` without any option argument (6.2.8 p. 33 or using `\Xsublinesepside`. But in the second case, it will change the separator only for line number in side, not for the footnotes.

### 4.4.2 Locking lineation

`\startlock` The `\startlock` command, used in running text, locks the line number at its current  
`\endlock` value, until you insert `\endlock`. It can tell for itself whether you are in a patch of line or sub-line numbering. One use for line-number locking is in printing poetry: there the line numbers should be those of verse lines rather than of printed lines, even when a verse line requires several printed lines. But in this case you may use the `\stanza` mechanism, see 8 p. 41.

`\lockdisp` When line-number locking is used, several printed lines may have the same line number, and you have to specify whether you want the number attached to the first printed line or the last, or whether you just want the number printed by them all, assuming that the settings of the previous parameters requires the display of a line number for this line. You specify your preference using `\lockdisp{<arg>}`; its argument is a word, either `first`, `last`, or `all`. The package initially sets this as `\lockdisp{first}`.

### 4.4.3 Setting and changing line number

`\setline` In some cases you may want to modify the line numbers that are automatically cal-  
`\advanceline`

culated: if you are printing only fragments of a work but want to print line numbers appropriate to a complete version, for example. The `\setline{<num>}` and `\advanceline{<num>}` commands may be used to change the current line's number (or the sub-line number, if sub-lineation is currently on). They change both the marginal line numbers and the line numbers passed to the notes. `\setline` takes one argument, the value to which you want the line number set; it must be 0 or greater. `\advanceline` takes one argument, an amount that should be added to the current line number; it may be positive or negative.

`\setlinenum` The `\setline` and `\advanceline` macros should only be used within a `\pstart...pend` group. The `\setlinenum{<num>}` command can be used outside such a group, for example between a `\pend` and a `\pstart`. It sets the line number to `<num>`. It has no effect if used within a `\pstart...pend` group.

#### 4.4.4 Line number style

`\linenumberstyle` Line numbers are normally printed as arabic numbers. You can use `\linenumberstyle{<style>}`  
`\sublinenumberstyle` to change the numbering style. `<style>` must be one of:

`Alph` Uppercase letters (A ... Z).

`alph` Lowercase letters (a ... z).

`arabic` Arabic numerals (1, 2, ...)

`Roman` Uppercase Roman numerals (I, II, ...)

`roman` Lowercase Roman numerals (i, ii, ...)

Note that with the `Alph` or `alph` styles, 'numbers' must be between 1 and 26 inclusive.

Similarly `\sublinenumberstyle{<style>}` can be used to change the numbering style of sub-line numbers, which is normally arabic numerals.

#### 4.4.5 Skipping and hiding number

`\skipnumbering` When inserted into a numbered line the macro `\skipnumbering` causes the numbering of that particular line to be skipped; that is, the line number is unchanged and no line number will be printed. Note that if you use it in `\stanza`, you must call it at the beginning of the verse.

`\hidenumbering` 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.

#### 4.4.6 Execute code at each line

`\dolinehook` `\doinsidelinehook` `reledmac` provides an advanced feature for users. The argument passed to `\dolinehook{<arg>}` will be executed before slicing a new line in the paragraph. The argument passed to `\doinsidelinehook{<arg>}` will be executed before printing a new line. In many cases, the latter is more useful than the former. The file `examples/2-line_numbers_in_header.tex` provides an example for printing the first and last line numbers of a page in the header.

## 5 Apparatus commands

### 5.1 Terminology

We call “critical notes” notes which refer to both a lemma, that is a part of text and a line number. Critical notes are subdivided in critical footnotes and critical endnotes.

We call “familiar notes” notes which refer to a footnote mark in the main text.

`\reledmac` manages many series of notes of each category. A series of notes is identified by an uppercase letter. When the series letter is at the *beginning* of a command name, it refers to a critical footnote. When the series letter is at the *end* of a command name, it refers to a familiar footnote.

So :

- `\Afootnote` is a critical footnote of the series A.
- `\Bendnote` is a critical endnote of the series B.
- `\footnoteC` is a familiar footnote of the series C.

### 5.2 Critical notes

#### 5.2.1 The lemma

`\edtext` Within numbered paragraphs, all footnotes and endnotes are generated by the `\edtext` macro:

```
\edtext{<lemma>}{<commands>}
```

The `<lemma>` argument is the lemma in the main text: `\edtext` both prints this as part of the text, and makes it available to the `<commands>` you specify to generate notes.

For example:

```
I am happy :
I saw my friend \edtext{Smith}{
\Afootnote{Jones C, D.}}
on Tuesday.
```

1 I am happy : I saw my friend Smith on  
2 Tuesday.

---

1 Smith] Jones C, D.

The lemma `Smith` is printed as part of this sentence in the text, and is also made available to the footnote that specifies a variant, `Jones C, D`. The footnote macro is supplied with the line number at which the lemma appears in the main text.

The `<lemma>` may contain further `\edtext` commands. Nesting makes it possible to print an explanatory note on a long passage together with notes on variants for individual words within the passage. For example:

```
I am happy : \edtext{I saw my friend
\edtext{Smith}{\Afootnote{Jones
C, D.}} on Tuesday.}{
\Bfootnote{The date was
July 16, 1954.}}
}
```

1 I am happy : I saw my friend Smith on  
2 Tuesday.

---

1 Smith] Jones C, D.

---

1-2 I saw my friend Smith on Tuesday.] The  
date was July 16, 1954.

However, `\edtext` cannot handle overlapping but unnested notes—for example, one note covering lines 10–15, and another covering 12–18; an `\edtext` that starts in the `\lemma` argument of another `\edtext` must end there, too. (The `\lemma` and `\linenum` commands may be used to generate overlapping notes if necessary.)

### 5.2.2 Footnotes

The second argument of the `\edtext` macro, `\commands`, may contain a series of subsidiary commands that generate various kinds of notes.

`\Afootnote` Five separate series of the footnotes are maintained; each macro takes one argument  
`\Bfootnote` like `\Afootnote{\text}`. When all of the six are used, the A notes appear in a layer  
`\Cfootnote` just below the main text, followed by the rest in turn, down to the E notes at the bottom.  
`\Dfootnote` These are the main macros that you will use to construct the critical apparatus of your  
`\Efootnote` text.

If you need more series of critical notes, please look at 5.5.1 p. 29.

An optional argument can be added before the text of the footnote. Its value is a comma-separated list of options. The available options are:

- `fulllines` to disable `\Xtwolines` and `\Xmorethantwolines` features for this note (cf. 6.2.4 p. 31).
- `nonum` disables line numbering for this note. A horizontal blank space is added instead. You can use `\Xinplaceoflemmaseparator` to set it (6.5.1 p. 35).
- `nosep` to disable the lemma separator for this note.
- `linangesep=c` to change to `c` the separator between start line and end line for this particular note.

Example: `\Afootnote[nonum]{\text}`.

### 5.2.3 Endnotes

`\Aendnote` The package also maintains five separate series of endnotes.

`\Bendnote` If you do not need the endnotes facility, you should use `noend` option when loading  
`\Cendnote` `reledmac`.

`\Dendnote` The mechanism is similar to the one for footnotes: each macro takes one or more  
`\Eendnote` optional arguments and one single argument, like:

`\Aendnote[option]{\text}`.

`option` can contain a comma-separated list of values. Allowed values are:

- `fulllines` to disable `\Xendtwolines` and `\Xendmorethantwolines` features for this particular note (cf. 6.2.4 p. 31).
- `nonum` to disable line number for this particular note.
- `nosep` to disable the lemma separator for this particular note. A horizontal blank space is added instead. You can use `\Xendinplaceoflemmaseparator` to set it (6.5.2 p. 36).

- `linrangesep=<c>` to change to `<c>` the separator between start line and end line for this particular note.

`\doendnotes` Normally, endnotes are not printed: you must use the `\doendnotes{<s>}`, where `<s>` is the letter of the series to be printed. Put this command where you want the corresponding set of endnotes printed. In this case, all the endnotes of the `<s>` series are printed, for all numbered sections.

`\doendnotesbysection` However, you may want to print the endnotes of one given series covering the first numbered section, then the endnotes of another given series covering the first numbered section, then the endnotes of the first given series covering the second numbered section, then the endnotes of the second given series covering the second numbered section, and so forth. In this case, use `\doendnotesbysection{<s>}`. For each value of `<s>`, the first call of the command will print the notes for the first series, the second call will print the notes for the second series etc. For example, do:

```
\section{Endnotes}
\subsection{First text}
\doendnotesbysection{A}
\doendnotesbysection{B}
\subsection{Second text}
\doendnotesbysection{A}
\doendnotesbysection{B}
```

Note that by default inside endnotes no separator is used between the lemma and the content. However you can use the `\Xendlemmaseparator` macro to define one (6.5.2 p. 36).

As endnotes may be printed at any point in the document they always start with the page number where they are called.

#### 5.2.4 Paragraph in critical apparatus

By default, no paragraph can be made in the notes of critical apparatus. You can allow it by adding the options `parapparatus` when loading the package :

```
\usepackage[parapparatus]{eledmac}
```

Note that you *cannot* use paragraphs (e.g. blank lines or `\par`) inside of notes, when they are set to paragraph arrangement!

#### 5.2.5 Change lemma and line number

`\lemma` If you want to change the lemma that gets passed to the notes, you can do this by using `\lemma{<alternative>}` within the second argument to `\edtext` and before the note commands. The most common use of this command is to abbreviate the lemma that's printed in the notes. For example:



```

I am happy :
\edtext{I saw my friend
  \edtext{Smith}{\Afootnote{Jones
    C, D.}} on Tuesday.}
  {\lemma{I \dots\ Tuesday.}
  \Bfootnote{The date was
    July 16, 1954.}
}

```

1 I am happy : I saw my friend Smith on  
2 Tuesday.

---

1 Smith ] Jones C, D.

---

1-2 I ... Tuesday.] The date was July 16, 1954.

`\linenum` You can use `\linenum{⟨arg⟩}` to change the line numbers passed to the notes. `⟨arg⟩` actually consist of seven parameters: the page, line, and sub-line number for the start of the lemma; the same three numbers for the end of the lemma; and the font specifier for the lemma. As the argument to `\linenum`, you specify those seven parameters in that order, separated by vertical bars (the `|` character). I.e.

```
\linenum{⟨start page⟩|⟨s. line⟩|⟨s. sub-l.⟩|⟨end p.⟩|⟨e. l.⟩|⟨e. sub-l.⟩|⟨font⟩|}
```

However, you can retain the value computed by `reledmac` for any number by simply omitting it; and you can omit a sequence of vertical bars at the end of the argument. For example, `\linenum{|||23}` changes only the ending page number of the current lemma.

This command does not change the marginal line numbers in any way; it just changes the numbers passed to the notes. Its use comes in situations that `\edtext` has trouble dealing with for whatever reason. If you need notes for overlapping passages that aren't nested, for instance, you can use `\lemma` and `\linenum` to generate such notes despite the limitations of `\edtext`. If the `⟨lemma⟩` argument to `\edtext` is extremely long, you may run out of memory; here again you can specify a note with an abbreviated lemma using `\lemma` and `\linenum`. The numbers used in `\linenum` need not be entered manually; you can use the 'x-' symbolic cross-referencing commands below (10 p. 45) to compute them automatically.

Similarly, being able to manually change the lemma's font specifier in the notes might be important if you were using multiple scripts or languages. The form of the font specifier is three separate codes separated by `/` characters, giving the family, series, and shape codes as defined within NFSS.

### 5.2.6 Changing the names of commands for critical apparatus

The commands for generating the apparatus have been given rather bland names, because editors in different fields have widely divergent notions of what sort of notes are required, where they should be printed, and what they should be called. But this does not mean you have to type `\Afootnote` when you would rather type something you find more meaningful, like `\variant`.

We recommend that you create a series of such aliases and use them instead of the names chosen here; all you have to do is put commands of this form at the start of your file:<sup>14</sup>

```
\newcommandx{\variant}[2][1,usedefault]{\Afootnote[#1]{#2}}
```

<sup>14</sup>We use `\newcommand` and `\newcommandx` instead of classical `\let` command because the `edtabular` environments have to modify the notes definition, and we need to use the newest definition of notes. Read the handbook of `xargs` to know more about `\newcommandx`.

```

\newcommandx{\explanatory}[2][1,usedefault]{\Bfootnote[#1]{#2}}
\newcommand{\trivial}[1]{\Aendnote{#1}}
\newcommandx{\testimonia}[2][1,usedefault]{\Cfootnote[#1]{#2}}

```

### 5.3 Disambiguation of identical words in the apparatus

Sometimes, the same word occurs twice (or more) in the same line. `reledmac` provides tools to disambiguate references in the critical notes. The lemma will be followed by a reference number if a given word occurs more than once in the same line.

#### 5.3.1 Basic use

`\sameword` To use this tool, you have to mark every occurrence of the potentially ambiguous term with the `\sameword` command:

```
Lupus \sameword{aut} canis \edtext{\sameword{aut}}{\Afootnote{et}} felix
```

In this example, `aut` will be followed, in the critical note, by the exponent 2 if it is printed in the same line as the first `aut`, but it will not if it is printed in a different line. The number is printed only after the second run.

#### 5.3.2 Notes about input encoding with UTF-8 processor

If you use UTF-8 processor, like `XYTeX` or `LuaTeX`, there should not be any glitches. However, pay attention to how characters are encoded. Similar-looking characters may be represented differently in unicode numbering.

For instance, in Greek, “α” has two possible unicode numbers:

- GREEK SMALL LETTER ALPHA (U+03B1) + COMBINING GREEK YPOGEGRAMMENI (U+0345)
- GREEK SMALL LETTER ALPHA WITH YPOGEGRAMMENI (U+1FB3)

Which unicode number you use depends, many times, on your keyboard configuration (the computer-input system).

Inside `reledmac`, the `\sameword` command considers these two unicodes (code positions) as different characters. If you use only one unicode number consistently, the distinction will probably make no difference to how your text looks, but `\sameword` will process the text inaccurately, based on the unicode numbers. To prevent this, do the following:

- If you use `XYTeX`, add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use `LuaTeX`, use the `unnormalize` package of Michal Hoftich<sup>15</sup> with the `buffer` option set to true.

<sup>15</sup><https://github.com/michal-h21/unnormalize>.

With these tools, Xe<sub>2</sub>TeX / LuaTeX will dynamically normalize unicode input when reading the file. Consequently, you will have no problems with the `\sameword` command.

### 5.3.3 Use with `\lemma` command

If you use the `\lemma` command, `reledmac` cannot know to which occurrence of `\sameword` in the first argument of `\edtext` a word marked with `\sameword` in `\lemma` should refer.

For example in the following example:

```
some thing
  \edtext{\sameword{sw}
    and other \sameword{sw}
    and again \sameword{sw}
    it is all}%
{\lemma{\sameword{sw} \ldots all}\Afootnote{critical note}}.%
```

`reledmac` cannot know if the “sw” in `\lemma` refers to the word after “thing”, after “other”, or after “again”.

Consequently, you must tell `reledmac` to which instance of `\sameword` you are referring in the first argument of `\edtext`:

- In the content of `\lemma`, use `\sameword` with no optional argument.
- In the first argument of `\edtext`, use `\sameword` with the optional argument [ $\langle X \rangle$ ].  $\langle X \rangle$  is the depth of the `\edtext` where the `\lemma` is used. So if the `\lemma` is called in a `\edtext` inside another `\edtext`,  $\langle X \rangle$  is equal to 2. If the `\lemma` is called in a `\edtext` “of first level”,  $\langle X \rangle$  is equal to 1. If the lemma is called in both 1 and 2 `\edtext` depth,  $\langle X \rangle$  is 1, 2. If that word is referenced in the lemma of every `\edtext` depth,  $\langle X \rangle$  can also be set to `inlemma`.

Note that only words that are actually referenced in a `\lemma` need the optional argument. Therefore, the first `\sameword` in the example above should have “1” as its optional argument, to be referenced correctly in the lemma.

Note also that the  $\langle X \rangle$  does not refer to the level where the `\sameword` occurs, but to the level of the `\lemma` that refers to that `\sameword`. For example:

```
\edtext{some \edtext{\sameword[1]{word}}\Afootnote{om. M}}
  and other \sameword{word}
  and again a \sameword{word}
  it is all}%
{\lemma{some \sameword{word} \ldots all}\Afootnote{critical note}}.%
```

Here the `\sameword` occurs in an `\edtext` of level 2, but since it is referenced by `\lemma` on level 1, it has “1” in the optional argument.

In the following example figure, each framed box represents an `\edtext` level. Each number is an occurrence of `\sameword`. After a framed box, the text in superscript

represents the content of `\lemma` for that `\edtext` level. The text in subscript at the right of a number represents the content of the optional argument of `\sameword`.

$$\boxed{1_{\text{inlemma}} \boxed{2} 3_2^{1\dots 3} 4 5_1^{1\dots 5}}$$

The `\sameword` number 3 is called in a `\lemma` related to an `\edtext` of level 2. It must be marked by “2”.

The `\sameword` number 5 is called in a `\lemma` related to `\edtext` of level 1. It must be marked by “1”.

The `\sameword` number is called in two `\lemmas`: one related to a `\edtext` of level 1, the other related to `\edtext` of level 2. It must be marked by “1,2”. However, as `\lemma` is called only in level 1 and 2, “1,2” could be replaced by “inlemma”.

The `\sameword` number “2” is in the first argument of a `\edtext` of level 3, but it has no `\lemma`-command, so there is no need to mark it.

### 5.3.4 Customizing

`\showwordrank` You can redefine the `\showwordrank` macro to change the way the number is printed. The default value is

```
\newcommand{\showwordrank}[2]{%
  #1\textsuperscript{#2}%
}
```

## 5.4 Familiar notes

### 5.4.1 Basic use

`\footnoteA` As well as the standard L<sup>A</sup>T<sub>E</sub>X footnotes generated via `\footnote`, the package also provides five series of additional footnotes called `\footnoteA` through `\footnoteE`. These  
`\footnoteB` provide the familiar marker in the text, and the marked text at the foot of the page can be  
`\footnoteC` formatted using any of the styles described for the critical footnotes. Note that the ‘regular’  
`\footnoteD` footnotes have the series letter at the end of the macro name whereas the critical  
`\footnoteE` footnotes have the series letter at the start of the name.

### 5.4.2 Customizing mark

`\thefootnoteA` Each series uses a set of macros for styling the marks. The mark numbering scheme of  
`\bodyfootmarkA` series A is defined by the `\thefootnoteA` macro; the default is:  
`\footfootmarkA` `\renewcommand*{\thefootnoteA}{\arabic{footnoteA}}`

The appearance of the mark in the text is controlled by `\bodyfootmarkA` which is defined as:

```
\newcommand*{\bodyfootmarkA}{%
  \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmarkA}}}}
```

The command `\footfootmarkA` controls the appearance of the mark at the start of the footnote text. It is defined as:

```
\newcommand*{\footfootmarkA}{\textsuperscript{\@nameuse{@thefnmarkA}}}
```

There are similar command triples for the other series.

### 5.4.3 Separator for multiple footnotes

The `footmisc` package [Fai03] by Robin Fairbairns has an option whereby sequential footnote marks in the text can be separated by commas<sup>3,4</sup> like so. As a convenience `reledmac` provides this automatically.

`\multfootsep` `\multfootsep` is used as the separator between footnote markers. Its default definition is:  
`\providecommand*\multfootsep{\textsuperscript{\normalfont,}}`  
 and can be changed if necessary.

## 5.5 Changing series

### 5.5.1 Create a new series

If you need more than five series of critical footnotes, you can create extra series, using `\newseries` command. For example, to create F and G series `\newseries{G,H}`.

### 5.5.2 Delete series

As the number of series which are defined increases, `reledmac` gets slower. If you do not need all of the six standard series (A–E), you can load the package with the `series` option. For example if you need only series A and B, use:

```
\usepackage[series={A,B}]{eledmac}
```

### 5.5.3 Series order

The default series order is the one called with the `series` option of the package, or, if this option is not used, A, B, C, D, E. Series order determines footnotes order.

`\seriesatbegin` `\seriesatend` However in some specific cases, you need to change the series order at some point inside the document. You can use `\seriesatbegin{<s>}` to pull up a given series `<s>` to the beginning, or `\seriesatend{<s>}` to push it down to the end.

## 5.6 Position of critical and familiar footnotes

`\fnpos` `\mpfnpos` There is a historical incoherence in `(r)(e)ledmac`. The familiar footnotes are before the critical footnotes in a normal page, but after in a minipage or in a ledgroup. However, it is possible to change the relative position of both types of footnotes. If you want to have familiar footnotes after critical footnotes in a normal page, use:

```
\fnpos{critical-familiar}
```

Or, if you want a minipage or ledgroup to have critical footnotes after familiar footnotes, use:

```
\mpfnpos{familiar-critical}
```

## 6 Critical apparatus appearance

Some commands can be used to change the display of the footnotes. All can have an optional argument [ $\langle s \rangle$ ], which is the letter of the series — or a list of letters separated by comma — depending on which option is applied. If the optional argument is omitted or empty, the setting will apply to the entire series.

When a length, noted  $\langle l \rangle$ , is used, it can be stretchable:  $a$  plus  $b$  minus  $c$ . The final length  $m$  is calculated by  $\text{\LaTeX}$  to have:  $a - c \leq m \leq a + b$ . If you use some relative unit<sup>16</sup>, it will be relative to font size of the footnote, except for commands concerning the place kept by the notes — including blank space.

Some commands are boolean, indicating when an option is enabled. If you want to disable the option after enabling it, you must use [`false`] as the second optional argument. For example:

- `\XX[A] [false]` to disable the ‘XX’ option for the series A.
- `\XX[] [false]` to disable it for all series.

There is also name convention:

- Names prefixed by X are for setting of critical footnotes.
- Names prefixed by Xend are for setting of critical endnotes.
- Names suffixed by X are for setting of familiar footnotes.

### 6.1 Notes arrangement in a series

`\Xarrangement`  
`\arrangementX`

By default, all footnotes are formatted as a series of separate paragraphs in one column. Three other formats are also available for notes.

Use `\Xarrangement[\langle s \rangle]{\langle a \rangle}` to change the arrangement of the  $\langle s \rangle$  series of critical footnotes and `\arrangementX[\langle s \rangle]{\langle a \rangle}` to change the arrangement of the  $\langle s \rangle$  series of familiar footnotes.

The value of  $\langle a \rangle$  can be one of the following

- `paragraph` formats all the footnotes of a series as a single paragraph. If you use this arrangement, you are strongly encouraged to read 18.1.5 p. 61.
- `twocol` formats them as separate paragraphs, but in two columns;
- `threecol`, in three columns.
- `normal`, restore normal arrangement.

You should set up the page layout parameters, and in particular the `\baselineskip` of the footnotes, before you call this macro because its action depends on these; too much or too little space will be allotted for the notes on the page if these macros use the wrong values.

<sup>16</sup>Like `em` which is the width of an ‘m’ in a given font.

Note that you *cannot* use paragraphs (e.g. blank lines or `\par`) or line breaks (`\break` or `\linebreak` or `\newline` etc.) inside of notes, when they are set to paragraph arrangement!

The notes arrangement must be called after having defined the document geometry setting. If you must change geometry setting inside your document, do not forget to call note arrangement again.

`\hspace` has been set for the pages that use this series of notes; otherwise  $\TeX$  will try to put too many or too few of these notes on each page. If you need to change the `\hspace` within the document, call the arrangement macro again afterwards to take account of the new value.

## 6.2 Control line number printing

### 6.2.1 Print line number only at first time

`\Xnumberonlyfirstinline` By default, the line number is printed in every note. If you want to print it only the first time for a given line number (i.e., one time for line 1, one time for line 2, etc.), you can use `\Xnumberonlyfirstinline[⟨s⟩]`.

`\Xnumberonlyfirstintwolines` Suppose you have a lemma on line 2 and a lemma between line 2 and line 3. With `\Xnumberonlyfirstinline`, the second lemma is considered to be on the same line as the first lemma. But if you use both `\Xnumberonlyfirstinline[⟨s⟩]` and `\Xnumberonlyfirstintwolines[⟨s⟩]`, a distinction is made.

`\Xsymmlinenum` For setting a particular symbol in place of the line number, you can use `\Xsymmlinenum[⟨s⟩]{⟨symbol⟩}` in combination with `\Xnumberonlyfirstinline[⟨s⟩]`. From the second lemma of the same line, the symbol will be used instead of the line number. Note that any command called in `⟨symbol⟩` must be robust. Use `\robustify` to robustify a non-robust command.

`\Xendnumberonlyfirstinline` For endnotes, `\Xendnumberonlyfirstinline`; `\Xendnumberonlyfirstintwolines` and `\Xendsymmlinenum` are the equivalents of `\Xnumberonlyfirstinline`; `\Xnumberonlyfirstintwolines` and `\Xsymmlinenum`.

### 6.2.2 Arbitrary text before line number

`\Xbeforenumber` `\Xbeforenumber[⟨s⟩]{⟨txt⟩}` allow to insert `⟨txt⟩` before the line number, only when the line number is printed, so taking into account `\Xnumberonlyfirstinline` and similar.

### 6.2.3 Separator for line range

`\Xlinerangeseparator` By default, the separator between the begin line and the end line in a lines' range is an en-dash in a normal font (`\textnormal{--}`). You can change it for critical footnotes with `\Xlinerangeseparator[⟨s⟩]{⟨text⟩}`, and with `\Xendlinerangeseparator[⟨s⟩]{⟨text⟩}` for critical endnotes.

### 6.2.4 Abbreviate line range

`\Xtwolines` If a lemma is printed on two subsequent lines, `reledmac` will print the first and the last  
`\Xmorethantwolines`

line numbers. Instead of this, it is also possible to print an abbreviation which stands for “line 1 and subsequent line(s)”.

To achieve this, use `\Xtwolines[s]{text}` and `\Xmorethantwolines[s]{text}`. The *text* argument of `\Xtwolines` will be printed if the lemma is on two lines, and the *text* argument of `\Xmorethantwolines` will be printed if the lemma is on three or more lines. For example:

```
\Xtwolines{sq.}
\Xmorethantwolines{sqq.}
```

will print “1sq.” for a lemma which falls on lines 1–2 and “1sqq.” for a lemma which falls on lines 1–4.

If you use `\Xtwolines` without setting `\Xmorethantwolines`, the *text* argument of `\Xtwolines` will be used for lemmas which fall on three or more lines.

However, if you want to use a short form (when the lemma overlaps two lines, but not more than two), use `\Xtwolinesbutnotmore[series]`.

When you use lineation by page, the final page number, if different from the initial page number, will not be printed, because the final page number is included in the `\Xendtwolines` symbol.

`\Xtwolinesonlyinsamepage`

However, you can force print the final page number with `\Xtwolinesonlyinsamepage[series]`.

You can disable `\Xtwolines` and related for a specific note by using the ‘[fullines]’ argument in the note macro cf. 5.2.2 p. 23.

`\Xendtwolines`  
`\Xendmorethantwolines`  
`\Xendtwolinesbutnotmore`

For endnotes, use these macros: `\Xendtwolines`; `\Xendmorethantwolines`; `\Xendtwolinesbutnotmore`; `\Xendtwolinesonlyinsamepage` instead of `\Xtwolines`; `\Xmorethantwolines`; `\Xtwolinesbutnotmore`; `\Xtwolinesonlyinsamepage`.

### 6.2.5 Disable line number

`\Xnonumber`  
`\Xendnonumber`

You can use `\Xnonumber[s]` if you do not want to have the line number in a footnote. `\Xendnonumber[s]` is the same for endnote.

### 6.2.6 Printing pstart number

`\Xpstart`

You can use `\Xpstart[s]` if you want to print the pstart number in the footnote, before the line and subline number. Note that when you change the lineation system, the option is automatically switched :

- If you use lineation by pstart, the option is enabled.
- If you use lineation by section or by page, the option is disabled.

`\Xpstarteverytime`

By default, the pstart number is printed only in the part of text where you have called `\numberpstarttrue`. We don’t know why you would like to print the pstart number in the notes and not in the main text. However, if you want to do it, you can call `\Xpstarteverytime[s]`. In this case, the pstart number will be printed every time in footnote.



`\Xonlypstart` In combination with `\Xpstart`, you can use `\Xonlypstart[⟨s⟩]` if you want to print only the pstart number in the footnote, and not the line and subline number.

### 6.2.7 Printing stanza number

`\Xstanza` You can use `\Xstanza[⟨s⟩]` if you want to print the stanza number in the footnote, before the line and subline number.

Of course the stanza number is printed only when you use `\numberstanza`

`\Xstanzaseparator`

When using `\Xstanza`, you can use `\Xstanzaseparator[⟨s⟩]{⟨text⟩}` to print `⟨text⟩` after the stanza number. Default value is empty.

### 6.2.8 Separator between line and subline numbers

`\Xsublinesep` `\Xsublinesep[⟨s⟩]{⟨txt⟩}` changes the separator between line and subline in footnotes.

**Employed without optional argument, it also change separator in side number.**

`\Xendsublinesep` `\Xendsublinesep[⟨s⟩]{⟨txt⟩}` does the same thing for endnotes.

**However, it does not change anything for the separator in side number. Use `\Xsublinesep` without optional argument or `\Xsublinesepside{⟨txt⟩}` to do it.**

The default value is `\textnormal{.}`.

### 6.2.9 Space around number

`\Xbeforenumber` With `\Xbeforenumber[⟨s⟩]{⟨l⟩}`, you can add some space before the line number in a footnote. If the line number is not printed, the space is not either. The default value is 0 pt.

`\Xafternumber` With `\Xafternumber[⟨s⟩]{⟨l⟩}` you can add some space after the line number in a footnote. If the line number is not printed, the space is not either. The default value is 0.5 em.

`\Xendbeforenumber` `\Xendafternumber` and `\Xendafternumber` are the equivalents of `\Xbeforenumber` and `\afternumber` for endnotes.

`\Xnonbreakableafternumber` By default, the space defined by `\Xafternumber` is breakable. With `\Xnonbreakableafternumber[⟨s⟩]` it becomes nonbreakable.

### 6.2.10 Space around line symbol

`\XbeforeSYMlinenum` With `\XbeforeSYMlinenum[⟨s⟩]{⟨l⟩}` you can add some space before the line symbol in a footnote. The default value is value set by `\Xbeforenumber`.

`\Xaftersymmlinum` With `\Xaftersymmlinum[⟨s⟩]{⟨l⟩}` you can add some space after the line symbol in a footnote. The default value is value set by `\Xafternumber`.

`\XendbeforeSYMlinenum` `\Xendaftersymmlinum` and `\Xendaftersymmlinum` are the equivalents of `\XbeforeSYMlinenum` and `\Xaftersymmlinum` for the endnotes.

### 6.2.11 Space in place of number

<code>\Xinplaceofnumber</code>	If no number or symbolic line number is printed, you can add a space, with <code>\Xinplaceofnumber [⟨s⟩]{⟨l⟩}</code> . The default value is 1 em.
<code>\Xendinplaceofnumber</code>	<code>\Xendinplaceofnumber [⟨s⟩]{⟨l⟩}</code> is the same, for critical endnotes.

### 6.2.12 Boxing line number and line symbol

<code>\Xboxlinenum</code>	It could be useful to put the line number inside a fixed box: the content of the note will be printed after this box. You can use <code>\Xboxlinenum [⟨s⟩]{⟨l⟩}</code> to do that. To subsequently disable this feature, use <code>\Xboxlinenum</code> with length equal to 0 pt. One use of this feature is to print line number in a column, and the note in an other column:  <pre>\Xhangindent{1em} \Xafternumber{0em} \Xboxlinenum{1em}</pre>
---------------------------	--

<code>\Xboxsymlinenum</code>	<code>\Xboxsymlinenum [⟨s⟩]{⟨l⟩}</code> is the same as <code>\Xboxlinenum</code> but for the line number symbol.
<code>\Xendboxsymlinenum</code>	<code>\Xendboxsymlinenum [⟨s⟩]{⟨l⟩}</code> is the same as <code>\Xboxsymlinenum</code> but for endnotes.
<code>\Xboxlinenumalign</code>	If you put line number in box, it will be aligned left inside the box. However, you can change it using <code>\Xboxlinenumalign [⟨s⟩]{⟨text⟩}</code> where <code>⟨text⟩</code> can be the following:  <b>L</b> to align left (default value);  <b>R</b> to align right;  <b>C</b> to center.

When using `\Xboxlinenum`, `reledmac` put all the line number description in the same box. That is, the same box will contain: the start line number, the dash, and either the end line number or the range symbol (like `ff.`). However, it is possible to box them in two different boxes.

- `\Xboxstartlinenum [⟨s⟩]{⟨l⟩}` will box the start line number in a box of length `⟨l⟩`. The content will be put at the right of the box.
- `\Xboxendlinenum [⟨s⟩]{⟨l⟩}` will box the dash plus the end line number or the range symbol in a box of length `⟨l⟩`. The content will be put at the left of the box.

With these two commands, it is possible to horizontally align the dash of line number when using critical notes, to obtain something like:

```
1
12-23
24ff.
```

<code>\Xendboxlinenum</code>	<code>\Xendboxlinenum [⟨s⟩]{⟨l⟩}</code> , <code>\Xendboxlinenumalign [⟨s⟩]{⟨text⟩}</code> , <code>\Xendboxstartlinenum [⟨s⟩]{⟨l⟩}</code>
<code>\Xendboxlinenumalign</code>	<code>\Xendboxendlinenum [⟨s⟩]{⟨l⟩}</code> are the same as, respectively, <code>\Xboxlinenum</code> and
<code>\Xendboxstartlinenumalign</code>	<code>\Xboxlinenumalign</code> , <code>\Xboxstartlinenum</code> , <code>\Xboxendlinenum</code> except in endnotes.
<code>\Xendboxendlinenumalign</code>	

### 6.3 For endnotes

`\Xendbeforepagenumber` `\Xendbeforepagenumber` [*s*] {*text*} defines the text before the page number in endnotes. Default value is p. (“p” followed by a dot).

`\Xendafterpagenumber` `\Xendafterpagenumber` [*s*] {*text*} defines the text after the page number in endnotes. Default value is ) (open parenthesis followed by a single space).

`\Xendlineprefixsingle` `\Xendlineprefixsingle` [*s*] {*text*} defines the text before the line number in endnotes, when there is only one line. Default value is empty.

`\Xendlineprefixmore` `\Xendlineprefixmore` [*s*] {*text*} defines the text before the line number in endnotes, when there is more than one line. Default value is empty. If you don’t define it, use the value defined by `\Xendlineprefixsingle`.

### 6.4 Arbitrary code around line number

`\Xendbhooklinenumber` `\Xendbhooklinenumber` [*s*] {*code*} is used to execute code before line number in endnotes. The code is executed before the `\Xendbeforelinenumber` space and before the `\Xendnotenumfont` font setting.

`\Xendahooklinenumber` `\Xendahooklinenumber` [*s*] {*code*} is used to execute code after line number in endnotes. The code is executed after the `\Xendafternumber` space.

`\Xendbhookinplaceofnumber` `\Xendbhookinplaceofnumber` [*s*] {*code*} is used to execute code before space or symbol which replace line number in endnotes. The code is executed before the `\Xendbeforemyslinenum` space and before the `\Xendnotenumfont` font setting.

`\Xendahookinplaceofnumber` `\Xendahookinplaceofnumber` [*s*] {*code*} is used to execute code after space or symbol which replace line number in endnotes. The code is executed after the `\Xendaftersynlinenum` space.

### 6.5 Separator between the lemma and the note

#### 6.5.1 For footnotes

`\Xlemmaseparator` By default, in a footnote, the separator between the lemma and the note is a right bracket (`\rbracket`)<sup>17</sup>. You can use `\Xlemmaseparator` [*s*] {*Xlemmaseparator*} to change it. The optional argument can be used to specify the series in which it is used. Note that there is a non-breakable space between the lemma and the separator, but a **breakable** space between the separator and the following text.

`\Xbeforelemmaseparator` Using `\Xbeforelemmaseparator` [*s*] {*l*} you can add some space between lemma and separator. If your lemma separator is empty, this space won’t be printed. The default value is 0 em.

`\Xafterlemmaseparator` Using `\Xafterlemmaseparator` [*s*] {*l*} you can add some space between separator and note. If your lemma separator is empty, this space will not be printed. The default value is 0.5 em.

`\Xnolemmaseparator` You can suppress the lemma separator, using `\Xnolemmaseparator` [*s*], which is simply a alias of `\Xlemmaseparator` [*s*] {}.

`\Xinplaceoflemmaseparator` With `\Xinplaceoflemmaseparator` [*s*] {*l*} you can add a space if no lemma separator is printed. The default value is 1 em.

<sup>17</sup>For polyglossia, when the lemma is RTL, the bracket automatically switches to a left bracket.

### 6.5.2 For endnotes

`\Xendlemmaseparator` By default, there is no separator inside endnotes between the lemma and the content of the note. You can use `\Xendlemmaseparator[⟨s⟩]{⟨Xendlemmaseparator⟩}` to change this. The optional argument can be used to specify the series in which it is used. A common value of `⟨Xendlemmaseparator⟩` is `\rbracket`.

Note that there is a non-breakable space between the lemma and the separator, but a **breakable** space between the separator and the following text.

`\Xendbeforelemmaseparator` Using `\Xendbeforelemmaseparator[⟨s⟩]{⟨l⟩}` you can add some space between the lemma and the separator. If your lemma separator is empty, this space won't be printed. The default value is 0 em.

`\Xendafterlemmaseparator` Using `\Xendafterlemmaseparator[⟨s⟩]{⟨l⟩}` you can add some space between the separator and the content of the note. If your lemma separator is empty, this space won't be printed. The default value is 0.5 em.

`\Xendinplaceoflemmaseparator` With `\Xendinplaceoflemmaseparator[⟨s⟩]{⟨l⟩}` you can add some space if you chose to remove the lemma separator. The default value is 0.5 em.

## 6.6 Font style

### 6.6.1 For line number

`\Xnotenumfont` `\Xnotenumfont[⟨s⟩]{⟨command⟩}` is used to change the font style for line numbers in critical footnotes ; `⟨command⟩` must be one (or more) switching command, like `\bfseries`.

`\Xendnotenumfont` `\Xendnotenumfont[⟨s⟩]{⟨command⟩}` is used to change the font style for line numbers in critical footnotes. `⟨command⟩` must be one (or more) switching command, like `\bfseries`.

`\notenumfontX` `\notenumfontX[⟨s⟩]{⟨command⟩}` is used to change the font style for note numbers in familiar footnotes. `⟨command⟩` must be one (or more) switching command, like `\bfseries`.

### 6.6.2 For the lemma

`\Xlemmadisablefontselection` By default, font of the lemma in footnote is the same as font of the lemma in the main text. For example, if the lemma is in italic in the main text, it is also in italic in note. The `\Xlemmadisablefontselection[⟨s⟩]` command allows to disable it for a specific series.

`\Xendlemmadisablefontselection` By default, font of the lemma in endnote is the same as font of the lemma in the main text. For example, if the lemma is in italic in the main text, it is also in italic in note. The command allows `\Xendlemmadisablefontselection[⟨s⟩]` to disable it for a specific series.

`\Xlemmafont` Use `\Xlemmafont[⟨s⟩]⟨cmd⟩` to apply a  $\TeX$  font command to the lemma. For example, to have boldface lemma:

`\Xendlemmafont`

```
\Xlemmafont{\bfseries}
```

`\Xendlemmafont⟨arg⟩⟨cmd⟩` is the same for endnotes.

### 6.6.3 For all notes

<code>\Xnotefontsize</code>	<code>\Xnotefontsize[⟨s⟩]{⟨command⟩}</code> is used to define the font size of critical footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .
<code>\notefontsizeX</code>	<code>\notefontsizeX[⟨s⟩]{⟨command⟩}</code> is used to define the font size of familiar footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .
<code>\Xendnotefontsize</code>	<code>\Xendnotefontsize[⟨s⟩]{⟨l⟩}</code> is used to define the font size of end critical footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .

## 6.7 Indent of notes content

<code>\Xparindent</code>	By default, <code>reledmac</code> does not add indentation before the paragraphs inside critical footnotes. Use <code>\Xparindent[⟨s⟩]</code> to enable indentation.
<code>\parindentX</code>	By default, <code>reledmac</code> does not add indentation before the paragraphs inside familiar footnotes. Use <code>\parindentX[⟨s⟩]</code> to enable indentation.
<code>\Xhangindent</code>	For critical notes NOT paragraphed you can define an indent with <code>\Xhangindent[⟨s⟩]{⟨l⟩}</code> , which will be applied in the second line of notes. It can help to make distinction between a new note and a break in a note. The default value is 0 pt.
<code>\hangindentX</code>	For familiar notes NOT paragraphed you can define an indentation with <code>\hangindentX[⟨s⟩]{⟨l⟩}</code> , which will be applied in the second line of notes. It can help to make a distinction between a new note and a break in a note.
<code>\Xendhangindent</code>	For critical endnotes NOT paragraphed you can define an indentation with <code>\Xendhangindent[⟨s⟩]{⟨l⟩}</code> , which will be applied in the second line of notes. It can help to make a distinction between a new note and a break in a note.

## 6.8 Arbitrary code at the beginning of notes

The three next commands add arbitrary code at the beginning of notes. As the name's space is local to the notes, you can use it to redefine some style inside the notes. For example, if you don't want the `pstart` number to be in bold, use :

```
\Xbhooknote{\renewcommand{\thepstart}{\arabic{pstart}.}}
```

<code>\Xbhooknote</code>	<code>\Xbhooknote[⟨s⟩]{⟨code⟩}</code> is to be used at the beginning of the critical footnotes.
<code>\bhooknoteX</code>	<code>\bhooknoteX[⟨s⟩]{⟨code⟩}</code> is to be used at the beginning of the familiar footnotes.
<code>\Xendbhooknote</code>	<code>\Xendbhooknote[⟨s⟩]{⟨code⟩}</code> is to be used at the beginning of the endnotes.

## 6.9 Options for footnotes in columns

### 6.9.1 Alignment

<code>\Xcolalign</code>	By default, text in footnotes of two or three columns are flush left and without hyphenation. However, you can change this with <code>\Xcolalign[⟨s⟩]{⟨code⟩}</code> for critical footnotes, and <code>\colalignX[⟨s⟩]{⟨code⟩}</code> for familiar footnotes.
<code>\colalignX</code>	

<code> must be one of the following command:

`\justifying` to have text justified, as usual with L<sup>A</sup>T<sub>E</sub>X. You can also let <code> empty.

`\raggedright` to have text left aligned, but *without hyphenation*. That is the default reledmac setting.

`\RaggedRight` to have text left aligned *with hyphenation* (requires ragged2e).

`\raggedleft` to have text right aligned, but *without hyphenation*.

`\RaggedLeft` to have text right aligned *with hyphenation* (requires ragged2e).

`\centering` to have text centered, but *without hyphenation*.

`\Centering` to have text centered *with hyphenation* (requires ragged2e).

### 6.9.2 Size of the columns

For the following four macros, be careful that the columns are made from right to left.

<code>\Xhsizetwocol</code>	<code>\Xhsizetwocol[⟨s⟩]{⟨l⟩}</code> is used to change width of a column when critical notes are displaying in two columns. Default value is <code>.45 \hsizetwocol</code> .
<code>\Xhsizethreecol</code>	<code>\Xhsizethreecol[⟨s⟩]{⟨l⟩}</code> is used to change width of a column when critical notes are displaying in three columns. Default value is <code>.3 \hsizethreecol</code> .
<code>\hsizetwocolX</code>	<code>\hsizetwocolX[⟨s⟩]{⟨l⟩}</code> is used to change width of a column when familiar notes are displaying in two columns. Default value is <code>.45 \hsizetwocolX</code> .
<code>\hsizethreecolX</code>	<code>\hsizethreecolX[⟨s⟩]{⟨l⟩}</code> is used to change width of a column when familiar notes are displaying in three columns. Default value is <code>.3 \hsizethreecolX</code> .

## 6.10 Options for paragraphed footnotes

### 6.10.1 Mark separation of notes

<code>\Xafternote</code>	You can add some horizontal space after a note by using <code>\Xafternote[⟨s⟩]{⟨l⟩}</code> (for critical footnotes) or <code>\afternoteX[⟨s⟩]{⟨l⟩}</code> (for familiar footnotes). The default value is <code>1em plus .4em minus .4em</code> .
<code>\afternoteX</code>	
<code>\Xparafootsep</code>	For paragraphed footnotes (see below), you can choose the separator between each note by using <code>\Xparafootsep[⟨s⟩]{⟨text⟩}</code> for critical notes and <code>\parafootsepX</code> for familiar notes. A common separator is the double pipe ( <code>  </code> ), which you can set by using <code>\Xparafootsep{<math>\parallel</math>}</code> .
<code>\parafootsepX</code>	

Note that if the symbol defined by `\Xsymlinenum` must be used at the beginning of a note, the `\Xparafootsep / \parafootsepX` is not used before this note.

### 6.10.2 Ragged text

<code>\Xragged</code>	Text in paragraphed critical notes is justified, but you can use <code>\Xragged[⟨s⟩]{L}</code> if you want it to be ragged left (i.e., right justified), or <code>\Xragged[⟨s⟩]{R}</code> if you want it to be ragged right (i.e., left justified).
<code>\raggedX</code>	Text in paragraphed footnotes is justified, but you can use <code>\raggedX[⟨s⟩]{L}</code> if you want it to be ragged left, or <code>\raggedX[⟨s⟩]{R}</code> if you want it to be ragged right.

## 6.11 Options for block of notes

### 6.11.1 Text before notes

`\Xtxtbeforenotes` You can add text before critical notes with `\Xtxtbeforenotes[⟨s⟩]{⟨text⟩}`.

### 6.11.2 Code before notes

`\Xbhookgroup` While `\Xtxtbeforenotes` is for typesetting code before notes, `\Xbhookgroup` and `\bhockgroupX` (respectively for critical and familiar) are for executing code before a groups of notes, between the rules and the printing of the notes.

### 6.11.3 Spacing

`\Xbeforenotes` You can change the vertical space before the rule of the critical notes with `\Xbeforenotes[⟨s⟩]{⟨l⟩}`. The default value is 1.2em plus .6em minus .6em.

**Be careful, the standard L<sup>A</sup>T<sub>E</sub>X footnote rule used by reledmac decreases by 3pt. This 3pt decrease is not changed by this command.**

`\beforenotesX` You can change the vertical space printed before the rule of the familiar notes with `\beforenotesX[⟨s⟩]{⟨l⟩}`. The default value is 1.2em plus .6em minus .6em.

**Be careful, the standard L<sup>A</sup>T<sub>E</sub>X footnote rule, which is used by reledmac, decreases 3pt. These 3pt are not changed by this command.**

`\preXnotes` You can set the space before the first series of critical notes printed on each page and set a different amount of space for each subsequent series on the page. You can do it with `\preXnotes{⟨l⟩}`. The default value is 0pt. You can disable this feature by setting the length to 0pt.

`\prenotesX` You can set the space before the first printed (in a page) series of familiar notes to be different from the space before other series. The default value is 0pt. You can do this with `\prenotesX{⟨l⟩}`. You can disable this feature by setting the length to 0pt.

### 6.11.4 Rule

`\Xafterrule` You can change the vertical space printed after the rule of the critical notes with `\Xafterrule[⟨s⟩]{⟨l⟩}`. The default value is 0pt.

**Be careful, the standard L<sup>A</sup>T<sub>E</sub>X footnote rule, which is used by reledmac, adds 2.6pt. These 2.6pt are not changed by this command.**

`\afterruleX` You can change the vertical space printed after the rule of the familiar notes with `\afterruleX[⟨s⟩]{⟨l⟩}`. The default value is 0pt.

**Be careful, the standard L<sup>A</sup>T<sub>E</sub>X footnote rule, which is used by reledmac, adds 2.6pt. These 2.6pt are not changed by this command.**

### 6.11.5 Maximum height

`\Xmaxhnotes` By default, one series of critical notes can take up to 80% of `\vsize`, before being broken to the next page. If you want to change the size use `\Xmaxhnotes[⟨s⟩]{⟨l⟩}`. Be careful : the length can't be flexible, and is relative to the the current font. For example, if you want the note to take, at most, 33% of the text height, do `\Xmaxhnotes{.33\textheight}`.



`\maxhnotesX` `\maxhnotesX[⟨s⟩]{⟨l⟩}` is the same as previous, but for familiar footnotes.

Note that in many cases, you should call these commands in the begin of the document, because the `\vsize` in the preamble is not the same as `\vsize` after the preamble. That why we recommend to you to add in your preamble

```
\AtBeginDocument{
  \maxhnotesX{0.8\textheight}
  \Xmaxhnotes{0.8\textheight}
}
```

Be careful with the two previous commands. Actually, for technical purposes, one paragraphed note is considered as one block. Consequently, it cannot be broken between two pages, even if you used these commands. The debug is in the `todolist`.

### 6.11.6 Width

`\Xwidth` `\Xwidth[⟨s⟩]{⟨l⟩}` sets the total width of critical footnotes. `\widthX[⟨s⟩]{⟨l⟩}` does the same for familiar footnotes.

`⟨l⟩` can be a length expression, parsable with `\dimexpr`. For example:

```
\Xwidth{\columnwidth+\marginparsep+\ledrsnotewidth}
\widthX{\columnwidth+\marginparsep+\ledrsnotewidth}
```

Note that changes the with of the block of notes. If you want to change the width of each column when typesetting notes in columns, use `\Xhsizetwocol`, `\Xhsizethreecol`, `\hsizetwocolX`, `\hsizethreecolX`, see 6.9.2 p. 38.

## 6.12 Footnotes and the `reledpar` columns

`\Xnoteswidthliketwocolumns` `\noteswidthliketwocolumnsX` If you use `reledpar \columns` macro, you can call :

- `\Xnoteswidthliketwocolumns[⟨s⟩]` to create critical notes with a two-column size width.
- `\noteswidthliketwocolumnsX[⟨s⟩]` to create familiar notes with a two-column size width.

## 6.13 Endnotes in one paragraph

`\Xendparagraph` By default, any new endnote starts a new paragraph. Use `\Xendparagraph[⟨s⟩]` to have all end notes of one given series set in one paragraph.

`\Xendafternote` You can add some space after a endnote series by using `\Xendafternote[⟨s⟩]{⟨l⟩}`. The default value is `1em plus .4em minus .4em`.

`\Xendsep` You can choose the separator between each note by `\Xendsep[⟨s⟩]{⟨text⟩}`. A common separator is the double pipe (`||`), which you can set by using `\Xendsep{${\parallel}$}`.



## 7 Fonts

One of the most important features of the appearance of the notes, and indeed of your whole document, will be the fonts used. We will first describe the commands that give you control over the use of fonts in the different structural elements of the document, especially within the notes, and then in subsequent sections specify how these commands are used.

For those who are setting up for a large job, here is a list of the complete set of `reledmac` macros relating to fonts that are intended for manipulation by the user: `\endashchar`, `\fullstop`, `\numlabfont`, and `\rbracket`.

`\numlabfont` Line numbers for the main text are usually printed in a smaller font in the margin. The `\numlabfont` macro is provided as a standard name for that font: it is initially defined as

```
\newcommand{\numlabfont}{\normalfont\scriptsize}
```

You might wish to use a different font if, for example, you preferred to have these line numbers printed using old-style numerals.

`\select@lemmafnt` We will briefly discuss `\select@lemmafnt` here because it is important to know about it now, although it is not one of the macros you would expect to change in the course of a simple job. Hence it is ‘protected’ by having the `@`-sign in its name.

When you use the `\edtext` macro to mark a word in your text as a lemma, that word will normally be printed again in your apparatus. If the word in the text happens to be in a font such as italic or bold you would probably expect it to appear in the apparatus in the same font. This becomes an absolute necessity if the font is actually a different script, such as Arabic or Cyrillic. `\select@lemmafnt` does the work of decoding `reledmac`’s data about the fonts used to print the lemma in the main text and calling up those fonts for printing the lemma in the note.

`\select@lemmafnt` is a macro that takes one long argument—the cluster of line numbers passed to the note commands. This cluster ends with a code indicating what fonts were in use at the start of the lemma. `\select@lemmafnt` selects the appropriate font for the note using that font specifier.

`reledmac` uses `\select@lemmafnt` in a standard footnote format macro called `\normalfootfmt`. The footnote formats for each of the layers A to E are `\let` equal to `\normalfootfmt`. So all the layers of the footnotes are formatted in the same way.

## 8 Verse

### 8.1 Basic

`\stanza` Use `\stanza` at the start of a stanza. Each line in a stanza is ended by an ampersand (`&`), and the stanza itself is ended by putting `\&` at the end of the last line.

### 8.2 Define stanza indents

`\stanzaindentbase` Lines within a stanza may be indented. The indents are integer multiples of the length `\stanzaindentbase`, whose default value is 20pt.

`\setstanzaindent` In order to use the stanza macros, **one must set the indentation values**. First the

value of `\stanzaindentbase` should be set, unless the default value 20pt is desired. Every stanza line indentation is a multiple of this.

To specify these multiples one invokes, for example `\setstanzaindent{3,1,2,1,2}`.

The numerical entries must be whole numbers, 0 or greater, separated by commas without embedded spaces. The first entry gives the hanging indentation to be used if the stanza line requires more than one print line.

If it is known that each stanza line will fit in one print line, then this first entry should be 0;  $\TeX$  does less work in this case, but no harm ensues if the hanging indentation is not 0 but is never used.

If you want the hanging verse to be flush right, you can use `\sethanginsymbol`: see p. 8.6 p. 43.

Enumeration is by stanza lines, not by print lines. In the above example the lines are indented one unit, two units, one unit, two units, with 3 units of hanging indentation in case a stanza line is too long to fit on one print line.

### 8.3 Repeating stanza indents

Since version 0.13, if the indentation is repeated every  $n$  verses of the stanza, you can define only the  $n$  first indentations, and indicate that they are repeated, defining the value of the `stanzaindentrepetition` counter at  $n$ . For example:

```
\setstanzaindent{5,1,0}
\setcounter{stanzaindentrepetition}{2}
```

is like

```
\setstanzaindent{5,1,0,1,0,1,0,1,0,1,0}
```

**Be careful: the feature is changed in `eledmac` 1.5.1. See Appendix A.3 p. 314.**

If you don't use the `stanzaindentrepetition` counter, make sure you have at least one more numerical entry in `\setstanzavalues` than the number of lines in the stanza.

If you want to disable this feature again, just put the counter to 0:

```
\setcounter{stanzaindentrepetition}{0}
```

The macros make no restriction on the number of lines in a stanza. Stanza indentation values (and penalty values) obey  $\TeX$ 's grouping conventions, so if one stanza among several has a different structure, its indentations (penalties) may be set within a group; the prior values will be restored when the group ends.

## 8.4 Manual stanza indent

`\stanzaindent`  
`\stanzaindent*`

You can set the indent of some specific verse by calling `\stanzaindent{⟨value⟩}` at the beginning of the verse, before any other character. In this case, the indent defined by `\setstanzaindent`s for this verse is skipped, and `{⟨value⟩}` is used instead.

If you use the mechanism of indent repetition, the next verse will be printed as it should be even if the current verse would have its normal indent value. In other words, using `\stanzaindent` in a verse does not shift the indent repetition.

However, if you want to shift the indent repetition, so the next verse has the indent normally used for the current verse, use `\stanzaindent*` instead of `\stanzaindent`.

## 8.5 Stanza breaking

`\setstanzapenalties`

When the stanzas run over several pages, it is often desirable that page breaks should arise between certain lines in the stanza, so a facility for including penalties after stanza lines is provided. If you are satisfied with the page breaks, you need not set the penalty values.

The command

```
\setstanzapenalties{1,5000,10100,5000,0}
```

results in a penalty of 5000 being placed after the first and third lines of the stanza, and a penalty of  $-100$  after the second.

The first entry “1” is a control value. If it is zero, then no penalties are passed on to  $\TeX$ , which is the default. Values between 0 and 10000 are penalty values; values between 10001 and 20000 have 10000 subtracted and the result is given as a negative penalty. The mechanism used for indentations and penalties requires unsigned values less than 32768. No penalty is placed after the last line, so the final `,0` in then example above could be omitted. A penalty of 10000 will prevent a page break; such a penalty is included automatically where there is stanza hanging indentation. A penalty of  $-10000$  (corresponding to the entry value 20000 in this context) forces a page break. Values in between act as suggestions as to the desirability of a page break at a given line. There is a subtle interaction between penalties and *glue*, so it may take some adjustment of skips and penalties to achieve the best results.

## 8.6 Hanging symbol

It is possible to insert a symbol in each line of hanging verse, as in French typography; for example, the opening bracket ‘[’. To insert it in `reledmac`, use macro `\sethangingsymbol{⟨h⟩}` with this code. In the example of French typography, do

`\sethangingsymbol`

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

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

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

## 8.7 Long verse and page break

If you want to prevent page breaks inside long verses, use the option `nopbinverse` when loading package, or use `\lednopbinversetrue`. Read 17.2 p. 59 for further details.

## 8.8 Content before/after verses

It is possible to add content, like a subtitle or a spacing, before or after verse:

- `\stanza` command can take a optional argument (in brackets). Its content will be printed before the stanza.
- `&` can be replaced by `\newverse` with two optional arguments (in brackets). The first will be printed after the current verse, the second before the next verse.
- `\&` can take a optional argument (in brackets). Its content will be printed after the stanza.

## 8.9 Numbering stanza

`\numberstanzatrue` If you want to automatically number stanzas, use `\numberstanzatrue`. In this case, the line number will restart at each `\stanza`.

`\numberstanzafalse`

If you want to disable this feature again, use `\numberstanzafalse`.

You can use this feature in combination with `\Xstanza` (6.2.7 p. 33).

`thestanza` . You can redefine `\thestanza` to change the aspect of stanza number. Default value is:

```
\renewcommand{\thestanza}{%
  \textbf{\arabic{stanza}}%
}
```

You can change the value of the `stanza` counter with the usual commands of  $\text{\LaTeX}$ .

`\stanzanumwrapper`

You can redefine `\stanzanumwrapper` in order to modify the way the stanza number is inserted in the flow of text. Default value is:

```
\newcommand{\stanzanumwrapper}[1]{%
  \flagstanza{#1}%
}
```

## 8.10 Various tools

`\ampersand` If you need to print an `&` symbol in a stanza, use the `\ampersand` macro, not `\&` which will end the stanza.

`\flagstanza` Putting `\flagstanza[⟨len⟩]{⟨text⟩}` at the start of a line in a stanza (or elsewhere) will typeset `⟨text⟩` at a distance `⟨len⟩` before the line. The default `⟨len⟩` is `\stanzaindentbase`.

## 8.11 Notes on empty lines

Since v2.3.0 of `reledmac`, empty lines when typesetting verses no longer produce new paragraphs, and consequently, do not insert vertical spaces. Use optional argument of `\stanza` or `\newverse` to insert vertical space (8.8 p. 44).

## 9 Grouping

In a `minipage` environment  $\LaTeX$  changes `\footnote` numbering from arabic to alphabetic and puts the footnotes at the end of the `minipage`.

`minipage` You can put numbered text with critical footnotes in a `minipage` and the footnotes are set at the end of the `minipage`.

You can also put familiar footnotes (see section 5.4) in a `minipage` but unlike with `\footnote` the numbering scheme is unaltered.

`ledgroup` `Minipages`, of course, are not broken across pages. Footnotes in a `ledgroup` environment are typeset at the end of the environment, as with `minipages`, but the environment includes normal page breaks. The environment makes no change to the `textwidth` so it appears as normal text; it just might be that footnotes appear in the middle of a page, with text above and below.

`ledgroup sized` The `ledgroup sized` environment is similar to `ledgroup` except that you must specify a width for the environment, as with a `minipage`.

`\begin{ledgroup sized}[<pos>]{<width>}`.

The required `<width>` argument is the text width for the environment. The optional `<pos>` argument is for positioning numbered text within the normal `textwidth`. It may be one of the characters:

l (left) numbered text is flush left with respect to the normal `textwidth`. This is the default.

c (center) numbered text is in the center of the `textwidth`.

r (right) numbered text is flush right with respect to the normal `textwidth`.

Note that normal text, footnotes, and so forth are all flush left.

`\begin{ledgroup sized}{\textwidth}` is effectively the same as `\begin{ledgroup}`

## 10 Cross referencing

The package provides a simple cross-referencing facility that allows you to mark places in the text with labels, and generate page and line number references to those places elsewhere using those labels.

### 10.1 Basic use

`\edlabel` First you place a label in the text using the command `\edlabel{<lab>}`. `<lab>` can be

almost anything you like, including letters, numbers, punctuation, or a combination—anything but spaces; you might type `\edlabel{toves-3}`, for example.<sup>18</sup>

`\edpageref`      Elsewhere in the text, either before or after the `\edlabel`, you can refer to its location via `\edpageref{<lab>}`, or `\edlineref{<lab>}` will produce, respectively, the page, line, sub-line and pstart on which the `\edlabel{<lab>}` command occurred.  
`\edlineref`  
`\sublineref`  
`\pstartref`

Note that the `\edlineref` command insert the side flag after the line number.

An `\edlabel` command may appear in the main text, or in the first argument of `\edtext`, but not in the apparatus itself. But `\edpageref`, `\edlineref`, `\sublineref`, `\pstartref` commands can also be used in the apparatus to refer to `\edlabels` in the text.

The `\edlabel` command works by writing macros to `ℒTeX.aux` file. You will need to process your document through `ℒTeX` twice in order for the references to be resolved.

You will be warned if you use `\edlabel{foo}` and `foo` has been used as a label before. The `ref` commands will return references to the last place in the file marked with this label. You will also be warned if a reference is made to an undefined label. (This will also happen the first time you process a document after adding a new `\edlabel` command: the auxiliary file will not have been updated yet.)

## 10.2 Cross-referencing to a critical note

If you want to refer to a word which is a lemma word, the `\edlabel` command should be in the first argument of `\edtext` command.

If you want to refer to the content of a `\Xfootnote`, the line and subline number printed will be the start line.

If you want to refer to starting and ending lines, you should use `\appref` and related tools (10.6.2 p. 48).

## 10.3 Cross-referencing which return a number in any case

`\xpageref`      Where #1 stands for the reference.  
`\xlineref`  
`\xsublineref`  
`\xpstartref`

However, there are situations in which you will want `reledmac` to return a number without displaying any warning messages about undefined labels or the like: if you want to use the reference in a context where `ℒTeX` is looking for a number, such a warning will lead to a complaint that the number is missing. This is the case for references used within the argument to `\linenum`, for example (see 5.2.5 p. 25).

For this situation, four variants of the reference commands, with the `x` prefix, are supplied: `\xpageref`, `\xlineref`, `\xsublineref` and `\xpstartref`. They have these limitations:

- They will not tell you if the label is undefined.
- They must be preceded in the file by at least one of the four other cross-reference commands—e.g., a `\edlabel{foo}` command, even if you never refer to that label—since those commands can all do the necessary processing of the `.aux` file, and the `\x . . .` ones cannot.

<sup>18</sup>More precisely, you should stick to characters in the `TeX` categories of “letter” and “other”.

- When `hyperref` is loaded, the `hyperref` link will not be added. (Indeed, it is not a limitation, but a feature.)
- With `reledpar`, the `\xlineref` does not insert the right side flag, in order to obtain a line number. Use `\xflagref` to obtain the side flag, depending of your flag.

### 10.3.1 Cross-referencing in order to define line number of a critical note

`\xxref` The macros `\xxref` and `\edmakelabel` let you manipulate numbers and labels in ways which you may find helpful in tricky situations.

The `\xxref{⟨lab1⟩}{⟨lab2⟩}` command generates a reference to a sequence of lines, for use in the second argument of `\edtext`. It takes two arguments, both of which are labels: e.g., `\xxref{mouse}{elephant}`. It calls `\linenum` (q.v., 5.2.5 p. 25 above) and sets the beginning page, line and subline numbers to those of the place where `\edlabel{mouse}` was placed, and the ending numbers to those where `\edlabel{elephant}` occurs.

## 10.4 Not automatic cross-referencing

`\edmakelabel` Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired—for example, if you want to refer to a page and line number in another volume of your edition. In such cases, you can use the `\edmakelabel{⟨lab⟩}{⟨numbers⟩}` macro so that you can ‘roll your own’ label.

For example, if you type `\edmakelabel{elephant}{10|25|0}` you will create a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. It is usually best to collect your `\edmakelabel` statements near the top of your document, so that you can see them at a glance.

## 10.5 Normal L<sup>A</sup>T<sub>E</sub>X cross-referencing

`\label` The normal `\label`, `\ref` and `\pageref` macros may be used within numbered text,  
`\ref` and operate in the familiar fashion.  
`\pageref`

## 10.6 References to start and end lines

### 10.6.1 Reference to main text lines

Many times, you may want to make a cross-reference to a passage that is defined by a start line and an end line. `reledmac` provides specific tools for this scenario.

`\edlabelS` Use `\edlabelS{⟨label⟩}` to mark the start line of the passage.

`\edlabelE` Use `\edlabelE{⟨label⟩}` to mark the end the end line of the passage. These two commands just create to label which are named `⟨label⟩:start` and `⟨label⟩:end`.

`\edlabelSE` Use `\edlabelSE{⟨label⟩}` to mark just one location in the text. Contrary to a classical `\edlabel`, the `⟨label⟩` could be use with `\SEref` and `\SErefwithpage`.

`\SEref` The main utility is to use them with three other commands. `\SEref{⟨label⟩}` will

make a cross-reference printed as a reference in critical footnotes.

`\Serefwithpage`     `\Serefwithpage` will make a cross-reference printed as a reference in critical endnotes.

`\Serefonlypage`     `\Serefonlypage` will make a cross-reference printed only with page number.

### 10.6.2 References to lines that are commented on in the apparatus

You may want to make a cross-reference to a passage that is referred to by `\edtext`. `reledmac` provides specific tools for this scenario.

`\applabel`     If you use `\applabel{<label>}` inside the second argument of a `\edtext`, `reledmac` will add a `\edlabel` at the beginning and end of the marked passage. The label at the beginning of the passage will have the title `<label>:start`, while the label at the end will have the title `<label>:end`.

If you use `\linenum` (5.2.5 p. 25) to refer to these labels, `reledmac` will use your line settings to refer to the passage.

`\appref`  
`\apprefwithpage`     You can also use `\appref{<label>}` and `\apprefwithpage{<label>}` to refer to these lines. The first one will print the lines as they are printed in the critical footnotes, while the second will print the lines as they are printed in endnotes.

### 10.6.3 Settings

`\setapprefprefixsingle`  
`\setapprefprefixmore`     **Specific to these tools** If you use `\apprefprefixsingle{<prefix>}`, `<prefix>` will be printed before the line numbers of a `\appref`-reference. If you use `\apprefprefixmore{<prefix>}`, `<prefix>` will be printed before the line numbers, if you refer to more than one line.

For example, you may use:

```
\setapprefprefixsingle{line~}
\setapprefprefixmore{lines~}
```

Note that if you have not used `\setapprefprefixmore` is empty, argument of `\setapprefprefixsingle` will be used in any case.

`\setSerefprefixsingle`  
`setSerefprefixmore`  
`\setSerefonlypageprefixsingle`  
`\setSerefonlypageprefixmore`     `\setSerefprefixsingle` and `\setSerefprefixmore` are similar for `\Seref` command.

Use `\setSerefonlypageprefixsingle{<prefix>}` to set the page prefix for `\Serefonlypage` when there is only one page. Use `\setSerefonlypageprefixmore{<prefix>}` to set it when there is more than one page. For example:

```
\setSerefonlypageprefixsingle{p.~}
\setSerefonlypageprefixmore{pp.~}
```

Note that if you do not use `\setSerefonlypageprefixmore`, the value of `\setSerefonlypageprefixsingle` is used instead.

**Also note that `\setSerefonlypageprefixsingle` is only a shortcut for `\Xendbeforepagenumber` (see 10.6.3 p. 49). So if you use `\Xendbeforepagenumber` without any optional argument, it will override this setting.**



**Linked to setting of critical endnotes and footnotes** Some commands who set the appearance of line numbers in critical footnotes also set the appearance of line numbers in `\appref` and `\SEref` if you call them *without the optional series argument*.

These commandes are the following:

- `\Xlineflag` (for `reledpar`), enabled by default.
- `\Xlinerangeseparator`
- `\Xmorethantwolines`
- `\Xsublinesep`
- `\Xtwolines`
- `\Xtwolinesbutnotmore`
- `\Xtwolinesonlyinsamepage`

If you want to make settings specific to `\appref` or `\SEref`, just call them with an optional argument containing a comma-separated list of command names (for example `appref,SEref`) or with a suffix equal to the command name (for example `appref`).

The same principle is available for `\apprefwithpage`, `\SErefwithpage` and `\SErefonlypage` with the following commands:

- `\Xendafterpagenumber` (not for `\SErefonlypage`)
- `\Xendbeforepagenumber`
- `\Xendlineflag` (for `reledpar`), enabled by default.
- `\Xendlineprefixmore`
- `\Xendlineprefixsingle`
- `\Xendlinerangeseparator`
- `\Xendmorethantwolines`
- `\Xendsublinesep`
- `\Xendtwolines`
- `\Xendtwolinesbutnotmore`
- `\Xendtwolinesonlyinsamepage`

**For one specific command** When calling `\appref` and `\SEref`, you can use as a first optional argument, in brackets (`[]`), any optional argument which can be used for critical footnotes (5.2.2 p. 23).

When calling `\apprefwithpage`, `\SErefwithpage` or `\SErefonlypage` you can use as a first optional argument, in brackets (`[]`), any optional argument which can be used for critical endnotes (5.2.3 p. 23).

## 10.7 Compatibility with xr package

The `\externaldocument` command of the `\xr` package allows making cross-references from an external document, with the standard  $\TeX$  commands `\label` and `\ref` (and related).

To use it with the `reledmac` cross-reference commands (i.e. `\edlabel` and related), you must do the following:

1. Load the `xr` package.
2. Load the `reledmac` package.
3. Use the `\externaldocument` document command.

## 11 Side notes

### 11.1 Basics

The `\marginpar` command does not work in numbered text. Instead, the package provides for non-floating sidenotes in either margin.

`\ledinnernote` `\ledinnernote{<text>}` will put `<text>` into the inner margin level with where the command was issued. Similarly, `\ledouternote` `\ledouternote{<text>}` puts `<text>` in the outer margin.

`\ledleftnote` `\ledsidenote{<text>}` will put `<text>` into the margin specified by the current setting of `\sidenotemargin{<location>}`. The permissible value for `<location>` is one out of the list `left`, `right`, `inner`, or `outer`, for example `\sidenotemargin{outer}`. The package's default setting is `\sidenotemargin{right}`

to typeset `\ledsidenotes` in the right hand margin. This is the opposite of the default margin for line numbers. The style for a `\ledsidenote` follows that for a `\ledleftnote` or a `\ledrightnote` depending on the margin it is put in.

If two note commands for the same side are called in the same line, they will be appended and separated by a comma.

### 11.2 Setting

#### 11.2.1 Width

`\ledlsnotewidth` The left sidenote text is put into a box of width `\ledlsnotewidth` and the right text into a box of width `\ledrsnotewidth`. These are initially set to the value of `\marginparwidth`.

#### 11.2.2 Vertical position

`\rightnoteupfalse` By default, sidenotes are placed to align with the last line of the note to which it refers. If you want they to be placed to align with the first line of the note to which it refers, use `\leftnoteupfalse` (for left note) and/or `\rightnoteupfalse` (for right note).

### 11.2.3 Distance to the main text

`\ledlsnotesep`    The texts are put a distance `\ledlsnotesep` (or `\ledrsnotesep`) into the left (or right) margin. These lengths are initially set to the value of `\linenumsep`.  
`\ledrsnotesep`  
`\ledlsnotefontsetup`    These macros specify how the sidenote texts are to be typeset. The initial definitions are:  
`\ledrsnotefontsetup`

```
\newcommand*\ledlsnotefontsetup{\raggedleft\footnotesize}% left
\newcommand*\ledrsnotefontsetup{\raggedright\footnotesize}% right
```

These can of course be changed to suit.

### 11.2.4 Separator between notes

`\setsidenotesep` If you have two or more sidenotes for the same line, they are separated by a comma. But if you want to change this separator, you can use `\setsidenotesep{<sep>}`.

## 12 Indexing

### 12.1 Basics

`\edindex`     $\TeX$  provides the `\index{<item>}` command for specifying that `<item>` and the current page number should be added to the raw index (`idx`) file. The `\edindex{<item>}` macro can be used in numbered text to specify that `<item>` and the current page & linenumber should be added to the raw index file.

Note that the file `.idx` will contain the right reference only after the third run, because of the internal indexing mechanism of `reledmac`. That means you must first run (Xe/Lua) $\TeX$  three times, then run `makeindex`, and then finally run (Xe/Lua) $\TeX$  again, in order to get an index with the right page numbers.

If the `imakeidx` or `indextools` package is used then the macro takes an optional argument, which is the name of a raw index file. For example `\edindex[line]{item}` will use `line.idx` as the raw file instead of `\jobname.idx`.

The minimal version of `imakeidx` package to be used is the version 1.3a uploaded on CTAN on 2013/07/11.

Be careful with the order of package loading and index declaration. You must use this order:

1. Load `imakeidx` or `indextools`.
2. Load `reledmac`.
3. Declare the index with the macro `\makeindex` of `imakeidx` and `indextools`.

### 12.2 Referring to critical notes

If you want to refer to a word inside an `\edtext{<lemma>}{<app>}` command, `\edindex` should be defined inside the first argument, e.g.,

```
The \edtext{creature\edlabel{elephant} was quite
unafraid}{\Afootnote{Of the mouse, that is.}}
```

If you add `\edindex` inside some `\Xfootnote` command, it will refer to that note, and a suffix *n* will be appended to the reference. You can redefine this suffix by redefining the command `\ledinnotemark`. Its actual definition is:

```
\newcommand{\ledinnotemark}[1]{#1\emph{n}}
```

### 12.3 Separator between page and line numbers

`\pagelinesep`

The page & linenummer combination is written as `page\pagelinesep line`, where the default definition is `\newcommand{\pagelinesep}{-}` so that an item on page 3, line 5 will be noted as being at 3-5. You can renew `\pagelinesep` to get a different separator.

- is the default separator used by the MAKEINDEX program.

Consequently, if you want to use an other `\pagelinesep`, you have to configure your `.ist` index style file. For example if you use `:` as separator<sup>19</sup>.

```
page_compositor ":"
delim_r ":"
```

Read the MAKEINDEX program's handbook about the `.ist` file.

### 12.4 Using xindy

Should you decide to use `xindy` instead of `makeindex` to transform your `.idx` files into `.ind` files, you must use some specific configuration file (`.xdy`) so that `xindy` can understand `eledmac` reference syntax of which the scheme is:

```
pagenumber-linenummer
```

An example of such a file is provided in the “examples” folder. Read the `xindy` handbook to learn how to use it.<sup>20</sup>

This file also provides, with an explanation, the settings that are needed to put `reledmac` lines numbers in parenthesis, in order to make a better distinction between line numbers and page ranges.

In any case, you must load `reledmac` with the `xindy` option, in order to generate a `.xdy` file which is specific to your document. This file is needed by the `.xdy` example file which is in the “examples” folder. Its default name is `reledmac-markup-attr.xdy`, but you can change it by using your own as an argument of the `xindy+hyperref` option.

If you chose to use both `xindy` and the `hyperref` package, you must do three more things:

<sup>19</sup>For further detail, you can read <http://tex.stackexchange.com/a/32783/7712>.

<sup>20</sup>Or, for people who read French, read <http://geekographie.maieul.net/174>.

1. Use `xindy+hyperref` option when loading the `reledmac` package. When you run (Xe/Lua)TeX with this option, a `.xdy` configuration file will be generated with all the settings needed to allow internal hyperlinking in each index entry which is created by `\edindex`.
2. Use `hyperindex=false` option when loading `hyperref`.
3. Uncomment — by removing the semicolons at the beginning of the relevant lines — some lines in the `<code>.xdy</code>` file provided in the “examples” folder in order to restore internal links in the index to be used by the standard `index` command.<sup>21</sup>.

## 12.5 Advanced setting

`\edindexlab` The `\edindex` process uses a `\label` and `\ref` mechanism to get the correct line number. It automatically generates labels of the form `\label{\edindexlab N}`, where `N` is a number, and the default definition of `\edindexlab` is:

```
\newcommand*\edindexlab}{\$&}
```

in the hopes that this will not be used by any other labels (`\edindex`'s labels are like `\label{\$&27}`). You can change `\edindexlab` to something else if you need to.

## 13 Glossary

`reledmac` provides mechanism to make glossaries with the `glossaries` package, referring not to the page, but to the page and line.

### 13.1 Preable setting

The standard compositor between page and line number in `reledmac` is a dash, while `glossaries` use, in standard, a dot. Consequently, you must:

- Or set `glossaries`:  
`\glsSetCompositor{-}`
- Or set `reledmac`:  
`\renewcommand{\pagelinesep}{-}`  
In this case, the will have consequences on your use of `\edindex`, and you should set your `.ist` file (?? p. ??).

### 13.2 Commands

The `\gls`, `\Gls`, and related commands of `glossaries` packages have a prefixed version with `ed`, which refers to the page line. The argument are the same as for the standard commands. So for example:

```
\edgls [options] {\label} [insert]
```

<sup>21</sup>These are the recommended lines to provide the best possible compatibility between `hyperref` and `xindy`, even without using `reledmac`.

## 14 Tabular material

L<sup>A</sup>T<sub>E</sub>X's normal tabular and array environments cannot be used where line numbering is being done; more precisely, they can be used but with odd results, so don't use them. However, `reledmac` provides some simple tabulation environments that can be line numbered. The environments can also be used in normal unnumbered text.

`edarrayl`      There are six environments; the `edarray*` environments are for math and `edtabular*`  
`edarrayc`      for text entries. The final `l`, `c`, or `r` in the environment names indicate that the entries  
`edarrayr`      will be flushleft (`l`), centered (`c`) or flushright (`r`). There is no means of specifying dif-  
`edtabularl`      ferent formats for each column, nor for specifying a fixed width for a column. The  
`edtabularc`      environments are centered with respect to the surrounding text.  
`edtabularr`

```
\begin{edtabularc}
1 & 2 & 3 \\
a & bb & ccc \\
AAA & BB & C
\end{edtabularc}
```

1	2	3
a	bb	ccc
AAA	BB	C

Entries in the environments are the same as for the normal array and tabular environments but there must be no ending `\\` at the end of the last row. *There must be the same number of column designators (the &) in each row.* There is no equivalent to any line drawing commands (such as `\hrule`). However, unlike the normal environments, the `ed...` environments can cross page breaks.

Macros like `\edtext` can be used as part of an entry.

For example:

```
\beginnumbering
\pstart
\begin{edtabularl}
\textbf{\Large I} & wish I was a little bug\edindex{bug} &
\textbf{\Large I} & eat my peas with honey\edindex{honey} \\
& With whiskers \edtext{round}{\Afootnote{around}} my tummy &
& I've done it all my life. \\
& I'd climb into a honey\edindex{honey} pot &
& It makes the peas taste funny \\
& And get my tummy gummy.\edindex{gummy} &
& But it keeps them on the knife.
\end{edtabularr}
\pend
\endnumbering
```

produces the following parallel pair of verses.

1		<b>I</b> wish I was a little bug		<b>I</b> eat my peas with honey
2		With whiskers round my tummy		I've done it all my life.
3		I'd climb into a honey pot		It makes the peas taste funny
4		And get my tummy gummy.		But it keeps them on the knife.

`\edtabcolsep`      The distance between the columns is controlled by the length `\edtabcolsep`.

`\spreadmath`     `\spreadmath{⟨math⟩}` typesets  $\langle math \rangle$  but the  $\langle math \rangle$  has no effect on the calculation of column widths. `\spreadtext{⟨text⟩}` is the analogous command for use in edtabular environments.

```

\begin{edarrayl}
1 & 2 & & 3 & & 4 & \\\
& \spreadmath{F+G+C} & & & & & \\
a & & bb & & ccc & & dddd \\
\end{edarrayl}

```

```

1 & 2 & 3 & 4 \\
F + G + C \\
a & bb & ccc & dddd

```

`\edrowfill`     The macro `\edrowfill{⟨start⟩}{⟨end⟩}{⟨fill⟩}` fills columns number  $\langle start \rangle$  to  $\langle end \rangle$  inclusive with  $\langle fill \rangle$ . The  $\langle fill \rangle$  argument can be any horizontal ‘fill’. For example `\hrulefill` or `\upbracefill`.



Note that every row must have the same number of columns, even if some would not appear to be necessary.

The `\edrowfill` macro can be used in both tabular and array environments. The typeset appearance of the following code is shown below.

```

\begin{edtabularr}
1 & & & & & & & & & & \\
Q & & & & & & & & & & \\
v & & & & & & & & & & \\
g & & & & & & & & & & \\
\edrowfill{1}{3}{\downbracefill} & & & & & & & & & & \\
k & & & & & & & & & & \\
1 & & & & & & & & & & \\
\end{edtabularr}

```

1	2	3	4		5
Q		fd	h		qwertziohg
v	wptz	x	y		vb
g	nnn				
			pq		dgh
k		l	co	ghweropjklmnbvcxys	
1	2	3			

You can also define your own ‘fill’. For example:

```

\newcommand*{\upbracketfill}{%
  \vrule height 4pt depth 0pt\hrulefill\vrule height 4pt depth 0pt}

```

is a fill like `\upbracefill` except it has the appearance of a (horizontal) bracket instead of a brace. It can be used like this:

```

\begin{edarrayc}
1 & 2 & & & & & & & & & \\
a & \edrowfill{2}{3}{\upbracketfill} & & & & & & & & & \\
A & B & & & & & & & & & \\
\end{edarrayc}

```

1	2	3	4
$a$	┌		$d$
$A$	$B$	$C$	$D$

`\edatleft`      `\edatleft[ $\langle math \rangle$ ]{ $\langle symbol \rangle$ }{ $\langle halfheight \rangle$ }` typesets the math  $\langle symbol \rangle$  as  $\left\langle symbol \right\rangle$  with the optional  $\langle math \rangle$  centered before it. The  $\langle symbol \rangle$  is twice  $\langle halfheight \rangle$  tall. The `\edatright` macro is similar and it typesets  $\right\langle symbol \right\rangle$  with  $\langle math \rangle$  centered after it.

```
\begin{edarrayc}
& 1 & 2 & 3 & \\
& 4 & 5 & 6 & \\
\edatleft[left =]{\}{1.5\baselineskip}
& 7 & 8 & 9 & \\
\edatright[= right]{\}{1.5\baselineskip}
\end{edarrayc}
```

$$left = \left( \begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array} \right) = right$$

`\edbeforetab`      `\edbeforetab{ $\langle text \rangle$ }{ $\langle entry \rangle$ }`, where  $\langle entry \rangle$  is an entry in the leftmost column, typesets  $\langle text \rangle$  left justified before the  $\langle entry \rangle$ . Similarly `\edaftertab{ $\langle entry \rangle$ }{ $\langle text \rangle$ }`, where  $\langle entry \rangle$  is an entry in the rightmost column, typesets  $\langle text \rangle$  right justified after the  $\langle entry \rangle$ .

For example:

```
\begin{edarrayl}
A & 1 & 2 & 3 \\
\edbeforetab{Before}{B} & 1 & 3 & 6 \\
C & 1 & 4 & \edaftertab{8}{After} \\
D & 1 & 5 & 0
\end{edarrayl}
```

Before	$A$	1	2	3	
	$B$	1	3	6	
	$C$	1	4	8	
	$D$	1	5	0	After

`\edvertline`      The macro `\edvertline{ $\langle height \rangle$ }` draws a vertical line  $\langle height \rangle$  high (contrast this with `\edatright` where the size argument is half the desired height).

`\edvertdots`

```
\begin{edarrayr}
a & b & C & d & & \\
v & w & x & y & & \end{edarrayr}
```



```
m & n & o & p & \\
k & & L & cvb & \edvertline{4pc}
\end{edarrayr}
```

<i>a</i>	<i>b</i>	<i>C</i>	<i>d</i>	
<i>v</i>	<i>w</i>	<i>x</i>	<i>y</i>	
<i>m</i>	<i>n</i>	<i>o</i>	<i>p</i>	
<i>k</i>		<i>L</i>	<i>cvb</i>	

The `\edvertdots` macro is similar to `\edvertline` except that it produces a vertical dotted instead of a solid line.

## 15 Sectioning commands

### 15.1 Sectioning commands without line numbers or critical notes

The standard sectioning commands (`\chapter`, `\section` etc.) can be used inside numbered text. In this case, you must call them as an optional argument of `\pstart` (4.2.3 p. 17):

```
\pstart[\section{section}]
Pstart content.
\pend
```

The line which contains them will not be numbered, and you cannot add critical notes inside.

### 15.2 Sectioning commands with line numbering and critical notes

You have to use the following commands:

- `\eledchapter[⟨text⟩]{⟨critical text⟩}`,
- `\eledchapter*`,
- `\eledsection[⟨text⟩]{⟨critical text⟩}`,
- `\eledsection*`,
- `\eledsubsection[⟨text⟩]{⟨critical text⟩}`,
- `\eledsubsection*`,
- `\eledsubsubsection[⟨text⟩]{⟨critical text⟩}`,
- `\eledsubsubsection*`.

These are equivalent to the  $\LaTeX$  commands. Each individual command must be called alone in a `\pstart ... \pend`:

```
\pstart
\eledsection*{xxxx\ledsidenote{section}}
\pend
\pstart
\eledsubsection*{xxxx\ledsidenote{sub}}
\pend
\pstart
normal text
\pend
```

After the first run, you will see only the text. This is normal. After the second run, you will see the formatting. Finally, with the third run, you will see the table of contents.

For technical reasons, the page break before `\elechapter` cannot be added automatically. You have to insert it manually via `\beforeeledchapter`, which must be called outside of a numbered section.

### 15.3 Optimization

`\noeledsec` If you are not going to have any `\eledxxx` commands, then load `reledmac` with `\noeledsec` option. That will suppress the generation of unneeded `.eledsec` files, save memory, and make `reledmac` run faster.

## 16 Quotation environments

The `quotation` and `quote` environments can be used so that the same definition/note appears both inside and outside a numbered section. The typographical consequences will resemble the outside numbered sections, based on the styles of the *book* class. However, if you use a package that redefines these environments, these redefinitions won't be available inside the numbered section. You must open any quotation environments inside a `\pstart ... \pend` block, not outside. A quotation environment **MUST NOT** be opened immediately after a `\pstart` and **MUST NOT** be closed immediately before a `\pend`.

In some cases, you do not want these environments to be redefined in numbered sections. You can load the package with the option `noquotation` to prevent this redefinition.

## 17 Page breaks

### 17.1 Control page breaking

`reledmac` and `reledpar` break pages automatically. However, you may sometimes want to either force page breaks, or prevent them. The packages provide two macros:

```
\ledpb
\lednopb
```

- `\ledpb` adds a page break.
- `\lednopb` prevents a page break, by adding one line to the current page if needed.

**These commands have effect only at the second run.**

`\ledpbsetting` These two commands take effect at the beginning of line in which they are called. For example, if you call `\ledpb` at l. 444, then l. 443 will be at the p.  $n$ , and the l. 444 at the p.  $n + 1$ . However, you can change the behavior and decide they will have effect after the end of the line, adding `\ledpbsetting{after}` at the beginning of your file (better: in your preamble). With the previous example, l. 444 will be on p.  $n$  and l. 445 will be on p.  $n + 1$ .

If you are using `reledpar` to typeset parallel pages, you must use `\lednopb` on both sides in the two corresponding lines. This is especially important when you are using stanzas; otherwise, the pages will be out of sync.

## 17.2 Prevent page break in a long verses

`\lednopbinversetrue` You can also decide to prevent page breaks between two lines of a long verse. To do this, use `nopbinverse` when loading package, or add `\lednopbinversetrue` in the beginning of your file (better: in your preamble).

This feature works only with verse of 2 lines and no more. It works on the third run, or on the fourth run if using `reledpar`. By default, when a long verse runs between two pages, a page break will be placed at the beginning of the verse. However, if you have added `\ledpbsetting{after}`, the page break will be placed at the end of the long verse and the page containing the long verse will have one extra line.

## 18 Miscellaneous

`\extensionchars` When the package assembles the name of the auxiliary file for a section, it prefixes `\extensionchars` to the section number. This is initially defined to be empty, but you can add some characters to help distinguish these files if you like; what you use is likely to be system-dependent. If, for example, you said `\renewcommand{\extensionchars}{!}`, then you would get temporary files called `jobname.!1`, `jobname.!2`, etc.

`\ifledfinal` The package can take options. The option ‘final’, which is the default is for final typesetting; this sets `\ifledfinal` to TRUE. The other option, ‘draft’, may be useful during earlier stages and sets `\ifledfinal` to FALSE.

`\showlemma` The lemma within the text is printed via `\showlemma{lemma}`. Normally, or with the ‘final’ option, the definition of `\showlemma` is:

```
\newcommand*\showlemma}[1]{#1}
```

so it just produces its argument. With the ‘draft’ option it is defined as

```
\newcommand*\showlemma}[1]{\textit{#1}}
```

so that its argument is typeset in an italic font, which may make it easier to check that all lemmas have been treated.

If you would prefer some other style, you could put something like this in the preamble:

```
\ifledfinal\else
  \renewcommand{\showlemma}[1]{\textbf{#1}}% or simply ...[1]{#1}
\fi
```

## 18.1 Known and suspected limitations

### 18.1.1 floatrow package compatibility

The floatrow package must be loaded before the reledmac.

### 18.1.2 ‘No room for a new’

Sometime, especially when using reledmac with other packages, you could obtain warning message such ‘no room for a new count’ or ‘no room for a new write’.

The first thing in order to prevent such problem is to use the options to optimize reledmac. For example, if you need only two series of notes, use `series={A,B}` option. Read 15.3 p. 58 in order to know which are there options.

However, if with these options you still have such message, here are some tricks.

‘**no room for a new count**’ is often caused by a conjunction with biblatex. Load reledmac (and reledpar) *before* biblatex.

‘**no room for a new write**’ can be caused by with multiple indexes. In this case, use `indextools` of imakeidx with the `splitindex` option, in order to obtain only one `.idx` file. If that does not solve your problem, you can use `morewrites` package. That should solve the problem, but  $\LaTeX$  will be slower.

If after reading and applying these advices you have still problem, contact us with a minimal working example.

### 18.1.3 Marginal notes

In general, reledmac’s system for adding marginal line numbers breaks anything that makes direct use of the  $\LaTeX$  insert system, which includes marginpars, footnotes and floats.

However, you can use both `\footnote` and the familiar footnote series notes in numbered text. A `\marginpar` in numbered text will throw away its contents and send a warning message to the terminal and log file, but will do no harm.

### 18.1.4 Paragraph shape

`\parshape` cannot be used within numbered text, except in a very restricted way.

`\ballast`  $\LaTeX$  is a three-pass system, but even after a document has been processed three times, there are some tricky situations in which the page breaks decided by  $\TeX$  never settle down. At each successive run, reledmac may oscillate between two different sets of page decisions. To stop this happening, should it arise, Wayne Sullivan suggested the inclusion of the quantity `\ballast`. The amount of `\ballast` will be subtracted from the penalties which apply to the page breaks calculated on the *previous* run through  $\TeX$ ,

thus reinforcing these breaks. So if you find your page breaks oscillating, insert `\setcounter{ballast}{100}` or some such figure, and with any luck the page breaks will settle down. Luckily, this problem does not crop up at all often.

### 18.1.5 Paragraphed footnotes

The restriction on explicit line-breaking in paragraphed footnotes, mentioned in a footnote 6.1 p. 30, and described in more detail on XII.6.3 p. 154, really is a nuisance if that is something you need to do. There are some possible solutions, described by Michael Downes, but this area remains unsatisfactory.

`\footfudgefiddle` For paragraphed footnotes  $\TeX$  has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say, to 68) to increase the estimate. You have to use `\renewcommand` for this, like:

```
\renewcommand{\footfudgefiddle}{68}
```

Note that you must call it *before* `\Xarrangement{paragraph}` or `\arrangementX{paragraph}`.

Any settings to ‘geometry’ must be made before `\Xarrangement / \arrangementX`.

Finally, in many cases you should use `\Xmaxhnotes` and / or `\maxhnotesX` (6.11.5 p. 39), in order to define the maximum height relative to `\textheight` and not to `\vsize`, because the `\vsize` value is not the same inside and outside of the preamble.

### 18.1.6 Use with other packages

Because of `reledmac`’s complexity, it may not play well with other packages. In particular `reledmac` is sensitive to commands in the arguments to the `\edtext` and `\*footnote` macros (this is discussed in more detail in section VI, and in particular the discussion about `\no@expands` and `\morenoexpands`). You will have to see what works or doesn’t work in your particular case.

`\morenoexpands` You can define the macro `\morenoexpands` to modify macros that you call within `\edtext`. Because of the way `reledmac` numbers the lines the arguments to `\edtext` can be processed more than once and in some cases a macro should only be processed once. One example is the `\colorbox` macro from the `color` package, which you might use like this:

```
... \edtext{\colorbox{mycolor}{lemma}}{\Afootnote{... \colorbox{...}}}
```

If you actually try this<sup>22</sup> you will find  $\TeX$  whinging ‘Missing { inserted’, and then things start to fall apart. The trick in this case is to specify either:

```
\newcommand{\morenoexpands}{\let\colorbox=0}
```

or

```
\makeatletter
```

<sup>22</sup>Reported by Dirk-Jan Dekker in the CTT thread ‘Incompatibility of “color” package’ on 2003/08/28.

```
\newcommand{\morenoexpands}{\let\colorbox\@secondoftwo}
\makeatother
```

(\@secondoftwo is an internal L<sup>A</sup>T<sub>E</sub>X macro that takes two arguments and throws away the first one.) The first incantation lets color show in both the main text and footnotes whereas the second one shows color in the main text but kills it in the lemma and footnotes. On the other hand if you use \textcolor instead, like

```
... \edtext{\textcolor{mycolor}{lemma}}{\Afootnote{... \textcolor{...}}}
```

there is no need to fiddle with \morenoexpands as the color will naturally be displayed in both the text and footnotes. To kill the color in the lemma and footnotes, though, you can do:

```
\makeatletter
\newcommand{\morenoexpands}{\let\textcolor\@secondoftwo}
\makeatother
```

It took Peter Wilson a little while to discover all this. If you run into this sort of problem you may have to spend some time experimenting before hitting on a solution.

If you want to use the option *bottom* of the `footmisc` package, you must load this package *before* the `reledmac` package.

### 18.1.7 Parallel typesetting

Peter Wilson has developed the `ledpar` package as an extension to `ledmac` specifically for parallel typesetting of critical texts. This also cooperates with the `babel` / `polyglossia` packages for typesetting in multiple languages. `reledpar` is the successor of the primitive `ledpar` package.

Peter Wilson also developed the `ledarab` package for handling parallel Arabic text in critical editions. However, this package is not maintained by Maïeul Rouquette. You should use the capabilities of a modern TeX processor, like Xe(La)TeX

## I Implementation overview

We present the `reledmac` code in roughly the order in which it is used during a run of  $\TeX$ . The order is *exactly* that in which it is read when you load The `Eledmac` package, because the same file is used to generate this manual and to generate the  $\LaTeX$  package file.

Most of what follows consists of macro definitions, but there are some commands that are executed immediately—especially at the start of the code. The documentation generally describes the code from the point of view of what happens when the macros are executed, though. As each macro is introduced, its name is printed in the margin.

After package options, we begin with the commands you use to start and stop line numbering in a section of text (Section II). Next comes the machinery for writing and reading the auxiliary file for each section that helps us count lines, and for creating list macros encoding the information from that file (Section V); this auxiliary file will be read at the start of each section, to create those list macros, and a new version of the file will be started to collect information from the body of the section.

Next are commands for marking sections of the text for footnotes (Section VI), followed by the macros that take each paragraph apart, attach the line numbers and insertions, and send the result to the vertical list (Section VII). The footnote commands (Section XII) and output routine (Section XXII) finish the main part of the processing; cross-referencing (Section XXIII) and endnotes (Section XIX) complete the story.

In what follows, macros with an `@` in their name are more internal to the workings of `reledmac` than those made up just of ordinary letters, just as in `PLAIN $\TeX$`  (see *The TeXbook*, p. 344). You are meant to be able to make free with ordinary macros, but the ‘`@`’ ones should be treated with more respect, and changed only if you are pretty sure of what you are doing.

## II Preliminaries

### II.1 Links with original `edmac`

Generally, these are the modifications to the original. `edmac` code:

- Replace as many `\def`’s by `\newcommand`’s as possible to avoid overwriting  $\LaTeX$  macros.
- Replace user-level  $\TeX$  counts by  $\LaTeX$  counters.
- Use the  $\LaTeX$  font handling mechanisms.
- Use  $\LaTeX$  messaging and file facilities.

### II.2 Package declaration

Announce the name and version of the package, which is targetted for `LaTeX2e`.

```

1 %<*code>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{reledmac}[2016/03/23 v2.9.0 typeset critical edition]%
4 %

```

### II.3 Package options

```

\ifledfinal Use this to remember which option is used, set and execute the options with final as the
\ifnocritical@ default. We use xkeyval in order to manage options with argument.
\if@noeled@sec \RequirePackage{xkeyval}
\ifnoend@ %
\ifnofamiliar@
\ifnoledgroup@ The parledgroup option is for reledpar. However, it has consequence on reledmac
\ifparapparatus@ internal command. So we need to define the boolean now.
\ifnoquotation@ \newif\ifparledgroup
\iflednopbinverse %
\ifparledgroup
\ifwidthliketwocolumns And now, the options of reledmac.
\ifxindy@ \DeclareOptionX{series}[A,B,C,D,E]{\xdef\default@series{#1}}
\ifxindyhyperref@ \ExecuteOptionsX{series}%
\ifeledmaccompat@
12 \newif\if@noeled@sec%
13 \DeclareOptionX{noeledsec}{\@noeled@sectrue}
14
15 \newif\ifnocritical@%
16 \DeclareOptionX{nocritical}{\nocritical@true}%
17
18 \newif\ifnofamiliar@%
19 \DeclareOptionX{nofamiliar}{\nofamiliar@true}%
20
21 \newif\ifnoledgroup@%
22 \DeclareOptionX{noledgroup}{\noledgroup@true}%
23
24 \newif\ifnoend@%
25 \DeclareOptionX{noend}{%
26 \let\l@dend@open\@gobble%
27 \let\l@dend@close\relax%
28 \global\let\l@dend@stuff=\relax%
29 \noend@true%
30 }%
31
32 \newif\ifnoquotation@
33 \DeclareOptionX{noquotation}{\noquotation@true}
34
35 \newif\ifledfinal
36 \DeclareOptionX{final}{\ledfinaltrue}
37 \DeclareOptionX{draft}{\ledfinalfalse}
38 \ExecuteOptionsX{final}

```



```

39
40 \newif\ifparapparatus@
41 \DeclareOptionX{parapparatus}{\parapparatus@true}
42
43 \newif\iflednopbinverse
44 \DeclareOptionX{nopbinverse}{\lednopbinversetrue}
45
46 \newif\ifwidthliketwocolumns%
47 \DeclareOptionX{widthliketwocolumns}{\widthliketwocolumnstrue}%
48
49 \newif\ifcontinuousnumberingwithcolumns
50 \DeclareOptionX{continuousnumberingwithcolumns}{\
continuousnumberingwithcolumnstrue}%
51
52 \newif\ifxindy@
53 \DeclareOptionX{xindy}[eledmac-markup-attr.xdy]{%
54   \AtBeginDocument{\immediate\openout\eledmac@xindy@out=#1}%
55   \newwrite\eledmac@xindy@out%
56   \xindy@true%
57   \gdef\eledmacmarkuplocdepth{:depth 1}%
58   \AtEndDocument{\immediate\closeout\eledmac@xindy@out}%
59 }%
60
61 \newif\ifxindyhyperref@
62 \DeclareOptionX{xindy+hyperref}{%
63   \xindyhyperref@true%
64 }%
65
66 \newif\ifeledmaccompat%
67 \DeclareOptionX{eledmac-compat}{%
68   \eledmaccompat@true%
69 }%
70 %
71 \ProcessOptionsX*\relax
72
73 %

```

We use the starred form of `\ProcessOptionsX` which executes options in the order listed in the source file: class options, then listed package options, so a package option can override a class option with the same name. This was suggested by Dan Luecking in the `ctt` thread *Class/package option processing*, on 27 February 2004.

## II.4 Loading packages

Loading package `xargs` to declare commands with optional arguments. `Etoolbox` is also used to make code clearer - for example, in dynamic command names (which can replace `\csname` etc.). Use `suffix` to declare commands with a starred version, `xstring` to work with strings, `ifluatex` and `ifxetex` to test if `LuaTeX` or `XYTeX` is running, and `ragged2e` to manage ragged justification for paragraphed notes.

```

74 \RequirePackage{xargs}
75 \RequirePackage{etoolbox}
76 \@ifl@t@r\fmtversion{2015/10/01}
77 {}%
78 {\RequirePackage{etex}%
79 \csname reserveinserts\endcsname{32}%
80 }
81 \RequirePackage{suffix}
82 \RequirePackage{xstring}
83 \RequirePackage{ifluatex}
84 \RequirePackage{ragged2e}
85 \RequirePackage{ifxetex}%
86 %

```

## II.5 Compatibility with Lua $\TeX$

Here, we enable some primitives for Lua $\TeX$ .

```

87 \ifx\directlua\undefined\else%
88   \directlua{tex.enableprimitives("",{"textdir","pardir","bodydir"})}
89 \fi
90 %

```

## II.6 Boolean flags

`\ifl@dmemoir` Define a flag for if the memoir class has been used.

```

91 \newif\ifl@dmemoir
92 \@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
93
94 %

```

`\if@ledgroup` Flag set to true inside a ledgroup environment.

```

95 \newif\if@ledgroup%
96 %

```

`\ifl@imakeidx` Define a flag for if the imakeidx package has been used.

```

97 \newif\ifl@imakeidx
98 \@ifpackageloaded{imakeidx}{\l@imakeidxtrue}{\l@imakeidxfalse}%False is the default value
99 %

```

`\ifl@indextools` Define a flag for if the indextools package has been used.

```

100 \newif\ifl@indextools%
101 \@ifpackageloaded{indextools}{%
102   \l@indextoolstrue%
103   \l@imakeidxtrue%

```

```

104 \let\imki@wrindexentry\indtl@wrindexentry%
105 }{}%
106 %

```

False is the default value. We consider `indextools` as a variant of `imakeidx`. That is why we set `\ifl@imakeidx` to true. We also let `\imki@wrindexentry` to `\indtl@wrindexentry`.

`\if@RTL` The `\if@RTL` is defined by the `bidi` package, which is sometimes loaded by *polyglossia*. But we define it as well if the `bidi` package is not loaded.

```

107 \ifdef{\if@RTL}{-}{\newif\if@RTL}
108 %

```

## II.7 Messages

All the messages are grouped here as macros. This saves  $\TeX$ 's memory when the same message is repeated and also lets them be edited easily.

`\reledmac@warning` Write a warning message.

```

109 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
110 %

```

`\reledmac@error` Write an error message.

```

111 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
112 %

```

```

\led@err@NumberingStarted13 \newcommand*{\led@err@NumberingStarted}{%
\led@err@NumberingNotStarted14 \reledmac@error{Numbering has already been started}{\@ehc}}
NumberingShouldHaveStarted15 \newcommand*{\led@err@NumberingNotStarted}{%
116 \reledmac@error{Numbering was not started}{\@ehc}}
117 \newcommand*{\led@err@NumberingShouldHaveStarted}{%
118 \reledmac@error{Numbering should already have been started}{\@ehc}}
119 %

```

```

\led@err@edtextoutsidestart20 \newcommand*{\led@err@edtextoutsidestart}{%
121 \reledmac@error{\string\edtext\space outside numbered paragraph (\pstart\
ldots\pend)}{\@ehc}}%
122 %

```

```

\led@mess@NotesChanged23 \newcommand*{\led@mess@NotesChanged}{%
124 \typeout{reledmac reminder: }%
125 \typeout{ The number of the footnotes in this section
126 has changed since the last run.}%
127 \typeout{ You will need to run LaTeX two more times

```

```

128 before the footnote placement}%
129 \typeout{ and line numbering in this section are
130 correct.}}
131 %

```

```

\led@mess@SectionContinued32 \newcommand*\led@mess@SectionContinued}[1]{%
133 \message{Section #1 (continuing the previous section)}}
134 %

```

```

\led@err@LineationInNumbered35 \newcommand*\led@err@LineationInNumbered}{%
136 \reledmac@error{You can't use \string\lineation\space within
137 a numbered section}{\@ehc}}
138 %

```

```

\led@warn@BadLineation39 \newcommand*\led@warn@BadLineation}{%
\led@warn@BadLinenummargin40 \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp41 \newcommand*\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp42 \reledmac@warning{Bad \string\linenummargin\space argument}}
143 \newcommand*\led@warn@BadLockdisp}{%
144 \reledmac@warning{Bad \string\lockdisp\space argument}}
145 \newcommand*\led@warn@BadSublockdisp}{%
146 \reledmac@warning{Bad \string\sublockdisp\space argument}}
147 %

```

```

\led@warn@NoLineFile48 \newcommand*\led@warn@NoLineFile}[1]{%
149 \reledmac@warning{Can't find line-list file #1}}
150 %

```

```

\led@warn@LineFileObsolete51 \newcommand*\led@warn@Obsolete}[1]{%
152 \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
153 %

```

```

\led@warn@BadAdvancelineSubline54 \newcommand*\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine55 \reledmac@warning{\string\advanceline\space produced a sub-line
156 number less than zero.}}
157 \newcommand*\led@warn@BadAdvancelineLine}{%
158 \reledmac@warning{\string\advanceline\space produced a line
159 number less than zero.}}
160 %

```

```

\led@warn@BadSetline61 \newcommand*\led@warn@BadSetline}{%
\led@warn@BadSetlinenum62 \reledmac@warning{Bad \string\setline\space argument}}
163 \newcommand*\led@warn@BadSetlinenum}{%
164 \reledmac@warning{Bad \string\setlinenum\space argument}}
165 %

\led@err@PstartNotNumbered66 \newcommand*\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart67 \reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumbered68 numbered section}{\@ehc}}
\led@err@PendNoPstart69 \newcommand*\led@err@PstartInPstart}{%
\led@err@AutoparNotNumbered70 \reledmac@error{\string\pstart\space encountered while another
\led@err@NumberingWithoutPstart71 \string\pstart\space was in effect}{\@ehc}}
172 \newcommand*\led@err@PendNotNumbered}{%
173 \reledmac@error{\string\pend\space must be used within a
174 numbered section}{\@ehc}}
175 \newcommand*\led@err@PendNoPstart}{%
176 \reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
177 \newcommand*\led@err@AutoparNotNumbered}{%
178 \reledmac@error{\string\autopar\space must be used within a
179 numbered section}{\@ehc}}
180 \newcommand*\led@err@NumberingWithoutPstart}{%
181 \reledmac@error{\string\beginnumbering...\string\endnumbering\space
without \string\pstart}{\@ehc}}%
182 %

\led@warn@BadAction83 \newcommand*\led@warn@BadAction}{%
184 \reledmac@warning{Bad action code, value \next@action.}}
185 %

\led@warn@DuplicateLabel86 \newcommand*\led@warn@DuplicateLabel}[1]{%
\led@warn@AppLabelOutEdtext87 \reledmac@warning{Duplicate definition of label `#1'\@gobble}%
\led@warn@RefUndefined88 \@latex@warning@no@line{Label `#1' multiply defined}%
\led@warn@RefUndefined89 }%
190 \newcommand*\led@warn@AppLabelOutEdtext}[1]{%
191 \reledmac@warning{\string\applabel\space outside of \string\edtext\space
`#1' on page \the\pageno.}}%
192 \newcommand*\led@warn@RefUndefined}[1]{%
193 \G@refundefinedtrue%
194 \reledmac@warning{Reference `#1' on page \the\pageno\space undefined.%
195 Using `000'.}%
196 \@latex@warning{Reference `#1' undefined\on@line}%
197 }%
198 \newcommand*\led@warn@pairRefUndefined}[1]{%
199 \G@refundefinedtrue%
200 \reledmac@warning{Reference `#1:start' and/or `#1:end' on page \the\
pageno\space undefined.

```

```

201         Using `??'.}%
202     \@latex@warning{Reference `#1:start' and/or `#1:end' undefined\on@line}%
203 }
204 %

```

```

\led@warn@NoMarginpars 205 \newcommand*\led@warn@NoMarginpars}{%
206     \reledmac@warning{You can't use \string\marginpar\space in numbered text
    }}
207 %

```

```

\led@warn@BadSidenotemargin 208 \newcommand*\led@warn@BadSidenotemargin}{%
209     \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
210 %

```

```

\led@warn@NoIndexFile 211 \newcommand*\led@warn@NoIndexFile}[1]{%
212     \reledmac@warning{Undefined index file #1}}
213 %

```

```

\led@warn@SeriesStillExist 214 \newcommand\led@warn@SeriesStillExist}[1]{%
215     \reledmac@warning{Series #1 is still existing !}%
216 }%
217 %

```

```

\led@err@ManySidenotes 218 \newcommand\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes 219     \ifledRcol{%
\led@err@ManyRightnotes 220         \reledmac@warning{\itemcount@\space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
221     \else%
222         \reledmac@warning{\itemcount@\space sidenotes on line \the\line@num\
space p. \the\page@num}%
223     \fi%
224 }%
225 \newcommand\led@err@ManyLeftnotes}{%
226     \ifledRcol{%
227         \reledmac@warning{\itemcount@\space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
228     \else%
229         \reledmac@warning{\itemcount@\space leftnotes on line \the\line@num\
space p. \the\page@num}%
230     \fi%
231 }%
232 \newcommand\led@err@ManyRightnotes}{%
233     \ifledRcol{%
234         \reledmac@warning{\itemcount@\space rightnotes on line \the\line@numR\
space p. \the\page@numR}%

```

```

235 \else%
236 \reledmac@warning{\itemcount@\space rightnotes on line \the\line@num\
space p. \the\page@num}%
237 \fi%
238 }%
239 %

```

```

\led@err@TooManyColumns40 \newcommand*\led@err@TooManyColumns}{%
\led@err@UnequalColumns41 \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn42 \newcommand*\led@err@UnequalColumns}{%
\led@err@HighEndColumn43 \reledmac@error{Number of columns is not equal to the number
\led@err@ReverseColumns44 in the previous row (or \protect\ \space forgotten?)}{\
@ehc}}
245 \newcommand*\led@err@LowStartColumn}{%
246 \reledmac@error{Start column is too low}{\@ehc}}
247 \newcommand*\led@err@HighEndColumn}{%
248 \reledmac@error{End column is too high}{\@ehc}}
249 \newcommand*\led@err@ReverseColumns}{%
250 \reledmac@error{Start column is greater than end column}{\@ehc}}
251 %

```

```

\led@err@EdtextWithoutFootnote52 \newcommand*\led@err@EdtextWithoutFootnote}{%
253 \reledmac@error{edtext without Xfootnote. Check syntaxis.}{\@ehc}%
254 }%
255 %

```

```

\led@err@FootnoteWithoutEdtext56 \newcommand*\led@err@FootnoteWithoutEdtext}{%
257 \reledmac@error{Xfootnote without edtext. Check syntax.}{\@ehc}%
258 }%
259 %

```

```

\led@err@ImakeidxAfterEledmac60 \newcommand*\led@err@ImakeidxAfterEledmac}{%
261 \reledmac@error{Imakeidx must be loaded before reledmac.}{\@ehc}%
262 }%
263 %

```

```

\led@err@IndextoolsAfterEledmac64 \newcommand*\led@err@IndextoolsAfterEledmac}{%
265 \reledmac@error{Indextools must be loaded before reledmac.}{\@ehc}%
266 }%
267 %

```

```

\led@err@fail@patch@@makecol68 \newcommand*\led@err@fail@patch@@makecol}{%
269 \reledmac@error{Fail to patch \string\@makecol\space command.}{\@ehc}%
270 }%
271 %

```

```

\led@error@fail@patch@@reinserts272 \newcommand{\led@error@fail@patch@@reinserts}{%
273   \reledmac@error{Fail to patch \string\@reinserts\space command.}\@ehc}%
274 }%
275 %

```

```

\led@error@fail@patch@@docclearpage276 \newcommand{\led@error@fail@patch@@docclearpage}{%
277   \reledmac@error{Fail to patch \string\@docclearpage\space command.}\@ehc}
278   %
279 }%
279 %

```

```

\led@error@fail@patch@@iiiminipage280 \newcommand{\led@error@fail@patch@@iiiminipage}{%
281   \reledmac@error{Fail to patch \string\@iiiminipage\space command.}\@ehc}
282   %
283 }%
283 %

```

```

\led@error@fail@patch@endminipage284 \newcommand{\led@error@fail@patch@endminipage}{%
285   \reledmac@error{Fail to patch \string\endminipage\space command.}\@ehc}%
286 }%
287 %

```

```

\led@warning@hsizeX@deprecated288 \newcommand{\led@warning@hsizeX@deprecated}{%
289   \reledmac@warning{\string\hsizeX\space command deprecated, use \string\
widthX\space instead.}%
290 }%
291 %

```

```

\led@warning@Xhsize@deprecated292 \newcommand{\led@warning@Xhsize@deprecated}{%
293   \reledmac@warning{\string\Xhsize\space command deprecated, use \string\
Xwidth\space instead.}%
294 }%
295 %

```

## II.8 Gobbling

Here, we define some commands which gobble their arguments.

```

\@gobblethree296 \providecommand*\@gobblethree}[3]{}
\@gobblefour297 \providecommand*\@gobblefour}[4]{}
\@gobblefive298 \providecommand*\@gobblefive}[5]{}
299 %

```



## II.9 Miscellaneous commands

`\showlemma` `\showlemma{lemma}` typesets the lemma text in the body. It depends on the option.

```

300 \ifledfinal
301   \newcommand*{\showlemma}[1]{#1}
302 \else
303   \newcommand*{\showlemma}[1]{\underline{#1}}
304 \fi
305
306 %

```

`\linenumberlist` The code for the `\linenumberlist` mechanism was given to Peter Wilson by Wayne Sullivan on 2004/02/11.

Initialize it as `\empty`.

```

307 \let\linenumberlist=\empty
308
309 %

```

`\@l@tempcnta` In imitation of  $\TeX$ , we create a couple of scratch counters.

`\@l@tempcntb`  $\TeX$  already defines `\@tempcnta` and `\@tempcntb` but Peter Wilson found in the past that it can be dangerous to use these (for example one of the AMS packages did something nasty to the `ccaption` package's use of one of these).

```

310 \newcount\@l@tempcnta \newcount\@l@tempcntb
311 %

```

## II.10 Prepare reledpar

`\ifl@dpairing` In preparation for the `reledpar` package, these are related to the 'right' text of parallel texts (when `\ifl@dpairing` is TRUE). They are explained in the `eledpar` manual.

`\ifl@dpaging`

`\ifl@dprintingpages`

`\ifl@dprintingcolumns`

`\ifpst@rtedL`

`\l@dnumpstartsL`

```

312 \newif\ifl@dpairing
313 \newif\ifl@dpaging%
314 \newif\ifl@dprintingpages%
315 \newif\ifl@dprintingcolumns%
316 \newif\ifpst@rtedL
317 \newcount\l@dnumpstartsL
318 %

```

`\ifledRcol` `\ifledRcol` is set to true in the Rightside environment. It must be not confused with `\ifledRcol@` which is set to true when a right line is processed, in `\Pages` or `\Columns`.

```

319 \newif\ifledRcol
320 \newif\ifledRcol@
321 %

```

`\ifnumberingR` The `\ifnumberingR` flag is set to true if we're within a right text numbered section.

```
322 \newif\ifnumberingR
323 %
```

The `\ifXnote@` macro is set to true when we are typesetting a critical footnote.

```
324 \newif\ifXnote@%
325 %
```

## II.11 Booleans provided by other optional packages which are required in any case

`\ifindtl@innote` The `\ifindtl@innote` and `\ifindtl@notenumber` are required even if `indextools` is not used.

```
326 \providebool{indtl@innote}%
327 \providebool{indtl@notenumber}%
328 %
```

## III Sectioning commands

`\section@num` You use `\beginnumbering` and `\endnumbering` to begin and end a line-numbered section of the text; the pair of commands may be used as many times as you like within one document to start and end multiple, separately line-numbered sections.  $\TeX$  will maintain and display a 'section number' as a count named `\section@num` that counts how many `\beginnumbering` and `\resumenumbers` commands have appeared; it need not be related to the logical divisions of your text.

`\extensionchars` Each section will read and write an associated 'line-list file', containing information used to do the numbering; the file will be called `\jobname.nn`, where `nn` is the section number. However, you may direct that an extra string be added before the `nn` in that filename, in order to distinguish these temporary files from others: that string is called `\extensionchars`. Initially it's empty, since different operating systems have greatly varying ideas about what characters are permitted in file names. So `\renewcommand{\extensionchars}{-}` gives temporary files called `jobname.-1`, `jobname.-2`, etc.

```
329 \newcount\section@num
330 \section@num=0
331 \let\extensionchars=\empty
332 %
```

`\ifnumbering` The `\ifnumbering` flag is set to true if we are within a numbered section (that is, between `\beginnumbering` and `\endnumbering`). You can use `\ifnumbering` in your own code to check whether you are in a numbered section, but do not change the flag's value.

`\numberingtrue`  
`\numberingfalse`

```

333 \newif\ifnumbering
334 %

```

`\beginnumbering` `\beginnumbering` begins a section of numbered text. When it is executed we increment the section number, initialize our counters, send a message to your terminal, and call macros to start the lineation machinery and endnote files.

The initializations here are trickier than they look. `\line@list@stuff` will use all of the counters that are zeroed here when it assembles the line-list and other lists of information about the lineation. But it will do all of this locally and within a group, and when it is done the lists will remain but the counters will return to zero. Those same counters will then be used as we process the text of this section, but the assignments will be made globally. These initializations actually apply to both uses, though in all other respects there should be no direct interaction between the use of these counters and variables in the two processing steps. For parallel processing :

- zero `\l@dnumpstartsL` – the number of chunks to be processed.
- set `\ifpstrtedL` to FALSE.

```

335 \newcommand*{\beginnumbering}{%
336   \ifnumbering
337     \led@err@NumberingStarted
338   \endnumbering
339   \fi
340   \global\numberingtrue
341   \global\advance\section@num \@ne
342   \initnumbering@reg
343   \message{Section \the\section@num }%
344   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
345   \l@dend@stuff
346   \setcounter{pstart}{1}
347   \ifl@dpairing
348     \global\l@dnumpstartsL \z@
349   \global\pstrtedLfalse
350 %

```

The tools for section's title commands are called:

- Define an empty list of pstart number where sectioning commands are called.
- Input auxiliary file with the description of section titles.
- Open the same auxiliary file to write in.

```

351 \else
352   \begingroup
353   \global\@afterindenttrue%In order to reestablish normal feature if the \
beginngroup was not here
354   \initnumbering@quote

```

```

355 \ifwidthliketwocolumns%
356 \csuse{setwidthliketwocolumns@\columns@position}%
357 \csuse{setpositionliketwocolumns@\columns@position}%
358 \fi%
359 \fi
360 \gdef\eled@sections@{}%
361 \if@noeled@sec\else%
362 \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@num}{-}{\
makeatother%
363 \immediate\openout\eled@sectioning@out=\jobname.eledsec\the\section@num
\relax%
364 \fi%
365 }
366 \newcommand*\initnumbering@reg}{%
367 \global\pst@rtedLfalse
368 \global\l@dnumpstartsL \z@
369 \global\absline@num \z@
370 \gdef\normal@page@break{}
371 \gdef\l@prev@pb{}
372 \gdef\l@prev@nopb{}
373 \global\line@num \z@
374 \global\subline@num \z@
375 \global\@lock \z@
376 \global\sub@lock \z@
377 \global\sublines@false
378 \global\let\next@page@num=\relax
379 \global\let\sub@change=\relax
380 \resetprevline@
381 \resetprevpage@num
382 }
383 %
384 %

```

`\endnumbering` `\endnumbering` must follow the last text for a numbered section. It takes care of notifying you when changes have been noted in the input that require running the file through again to move everything to the right place.

```

385 \def\endnumbering{%
386 \ifnumbering
387 \global\numberingfalse
388 \normal@pars
389 \ifnum\l@dnumpstartsL=0%
390 \led@err@NumberingWithoutPstart%
391 \fi%
392 \ifl@dpairing
393 \global\pst@rtedLfalse
394 \else
395 \ifx\insertlines@list\empty\else
396 \global\noteschanged@true
397 \fi

```

```

398     \ifx\line@list\empty\else
399         \global\noteschanged@true
400     \fi
401 \fi
402 \ifnoteschanged@
403     \led@mess@NotesChanged
404 \fi
405 \else
406     \led@err@NumberingNotStarted
407 \fi
408 \autoparfalse
409 \if@noeled@sec\else%
410     \immediate\closeout\eled@sectioning@out%
411 \fi%
412 \ifl@dpairing\else
413     \global\l@dnumpststartsL=\z@%
414     \endgroup
415 \fi
416 }
417 %

```

`\pausenumbering` The `\pausenumbering` macro is just the same as `\endnumbering`, but with the `\resumenummering` `\ifnumbering` flag set to true, to show that numbering continues across the gap.<sup>23</sup>

```

418 \newcommand{\pausenumbering}{%
419     \ifautopar\global\autopar@pausetrue\fi%
420     \endnumbering\global\numberingtrue}
421 %

```

The `\resumenummering` macro is a bit more involved, but not much. It does most of the same things as `\beginnumbering`, but without resetting the various counters. Note that no check is made by `\resumenummering` to ensure that `\pausenumbering` was actually invoked.

```

422 \newcommand*{\resumenummering}{%
423     \ifnumbering
424         \ifautopar@pause\autopar\fi
425         \global\pst@rtedLtrue
426         \global\advance\section@num \@ne
427         \led@mess@SectionContinued{\the\section@num}%
428         \line@list@stuff{\jobname.\extensionchars\the\section@num}%
429         \l@dend@stuff
430     \ifl@dpairing\else%
431         \begingroup%
432         \initnumbering@quote%
433         \ifwidthliketwocolumns%
434             \csuse{setwidthliketwocolumns@\columns@position}%
435             \csuse{setpositionliketwocolumns@\columns@position}%
436         \fi%

```

<sup>23</sup>Peter Wilson's thanks to Wayne Sullivan, who suggested the idea behind these macros.

```

437 \fi%
438 \ifcontinuousnumberingwithcolumns%
439   \ifdefined\line@numR%
440     \ifnum\line@numR>\line@num%
441       \expandafter\setlinenum\expandafter{\the\line@numR}%
442     \fi%
443     \ifnum\last@page@numR>\last@page@num%
444       \global\last@page@num=\last@page@numR%
445     \fi%
446   \fi%
447 \fi%
448 \else
449   \led@err@NumberingShouldHaveStarted
450   \endnumbering
451   \beginnumbering
452 \fi}
453
454
455 %

```

## IV List macros

We will make heavy use of lists of information, which will be built up and taken apart by the following macros; they are adapted from *The TeXbook*, pp. 378–379, which discusses their use in more detail.

These macros consume a large amount of the run-time of this code. We intend to replace them in a future version, and in anticipation of doing so have defined their interface in such a way that it is not sensitive to details of the underlying code.

The historical list tools of `ledmac` are kept, because in many cause there are more useful than `etoolbox`'s lists. They allows to get and delete the first element of a list in one operation. They also expands the items add to the list.

However, `etoolbox`'s lists are more useful to loop on them. Consequently, depending of what we need, we use one or either.

It could be nice to unify them to the  $\LaTeX$ 3 list, however such migration would take quite time with some risk of error, for a gain which will be minor.

`\list@create` The `\list@create` macro creates a new list. This macro does not do anything beyond initializing an empty list macro.

```

456 \newcommand*\list@create}[1]{%
457   \global\let#1=\empty%
458 }%
459 %

```

`\list@clear` The `\list@clear` macro just initializes a list to the empty list; it is no different from `\list@create` in its effect, but it is in its semantic .

```

460 \newcommand*{\list@clear}[1]{%
461   \global\let#1=\empty%
462 }
463 %

```

`\xright@appenditem` `\xright@appenditem` expands an item and appends it to the right end of a list macro. `\led@toksa` We want the expansion because we will often be using this to store the current value of a counter. `\xright@appenditem` creates global control sequences, like `\xdef`, and uses two temporary token-list registers, `\@toksa` and `\@toksb`.

```

464 \newtoks\led@toksa \newtoks\led@toksb
465 \global\led@toksa={\}
466 \long\def\xright@appenditem#1\to#2{%
467   \global\led@toksb=\expandafter{#2}%
468   \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
469   \global\led@toksb={}}
470 %

```

`\xleft@appenditem` `\xleft@appenditem` expands an item and appends it to the left end of a list macro; it is otherwise identical to `\xright@appenditem`.

```

471 \long\def\xleft@appenditem#1\to#2{%
472   \global\led@toksb=\expandafter{#2}%
473   \xdef#2{\the\led@toksa\expandafter{#1}\the\led@toksb}%
474   \global\led@toksb={}}
475 %

```

`\gl@p` The `\gl@p` macro removes the leftmost item from a list and places it in a control sequence. You type `\gl@p\l\to\z` (where `\l` is the list macro, and `\z` receives the left item). `\l` is assumed nonempty: use `\ifx\l\empty` to test for an empty `\l`. The control sequences created by `\gl@p` are all global.

```

476 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
477 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
478
479 %

```

## V Line counting

### V.1 Choosing the system of lineation

Line number can be reset at each section (default) ; at each page ; at each pstart. Here we define internal codes for these systems and the macros.

`\ifbypstart@` The `\ifbypage@` and `\ifbypstart@` flag specify the current lineation system:

```

\ifbypstart@
\byppstart@true
\byppstart@false
  \ifbypage@
    \bypage@true
    \bypage@false

```

- line-of-page: `bypstart@ = false` and `bypage@ = true`.
- line-of-pstart: `bypstart@ = true` and `bypage@ = false`.

reledmac will use the line-of-section system unless instructed otherwise.

```
480 \newif\ifbypage@
481 \newif\ifbypstart@
482 %
```

The `\ifbypage@R` and `\ifbypstart@R` flag specify the current lineation for right side in case of using `reledpar`. They are now defined because they are used in some specific code. `reledpar` will use the line-of-section system unless instructed otherwise.

```
\ifbypage@R83 \newif\ifbypage@R
\ifbypstart@R84 \newif\ifbypstart@R
485 %
```

`\lineation` `\lineation{<word>}` is the macro you use to select the lineation system. Its argument is a string: either page, section or pstart.

```
486 \newcommand*{\lineation}[1]{{
487 %
```

We can't change the lineation system inside numbering section.

```
488 \ifnumbering
489 \led@err@LineationInNumbered
490 \else
491 %
```

If the argument is page.

```
492 \def\@tempa{#1}\def\@tempb{page}%
493 \ifx\@tempa\@tempb
494 \global\bypage@true
495 \global\bypstart@false
496 \unless\ifnocritical@%
497 \Xpstart [] [false]%
498 \fi%
499 %
```

If the argument is pstart.

```
500 \else
501 \def\@tempb{pstart}%
502 \ifx\@tempa\@tempb
503 \global\bypage@false
504 \global\bypstart@true
505 \unless\ifnocritical@%
506 \Xpstart%
507 \fi%
508 %
```

And finally, if the argument is section (default).



```

509     \else
510         \def\@tempb{section}
511         \ifx\@tempa\@tempb
512             \global\bypage@false
513             \global\bystart@false
514             \unless\ifnocritical@%
515                 \Xpstart [] [false]%
516             \fi%
517 %

```

In other case, it is an error.

```

518     \else
519         \led@warn@BadLineation
520     \fi
521 \fi
522 \fi
523 \fi}}
524 %

```

## V.2 Line number margin

`\linenummargin` `\linenummargin{<word>}` specify which margin line numbers are in; it takes one argument, a string, which value can be `left`; `right`; `inner` or `outer`.  
`\line@margin` The selection is recorded in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.  
`\l@dgetline@margin`

```

525 \newcount\line@margin
526
527 \newcommand*\linenummargin}[1]{%
528     \l@dgetline@margin{#1}%
529     \ifnum\l@dtempcntb>\m@ne
530         \ifledRcol
531             \global\line@marginR=\l@dtempcntb
532             \led@warn@setting@in@rightside{\linenummargin}%
533         \else
534             \global\line@margin=\l@dtempcntb
535         \fi
536     \fi}}
537
538 \newcommand*\l@dgetline@margin}[1]{%
539     \def\@tempa{#1}\def\@tempb{left}%
540     \ifx\@tempa\@tempb
541         \l@dtempcntb \z@
542     \else
543         \def\@tempb{right}%
544         \ifx\@tempa\@tempb
545             \l@dtempcntb \@ne
546         \else
547             \def\@tempb{outer}%

```

```

548     \ifx\@tempa\@tempb
549         \l@dttempcntb \tw@
550     \else
551         \def\@tempb{inner}%
552         \ifx\@tempa\@tempb
553             \l@dttempcntb \thr@@
554         \else
555             \led@warn@BadLinenummargin
556             \l@dttempcntb \m@ne
557         \fi
558     \fi
559 \fi
560 \fi}
561
562 %

```

### V.3 Line number initialization and increment

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

```

563 \newcounter{firstlinenum}
564     \setcounter{firstlinenum}{5}
565 \newcounter{linenumincrement}
566     \setcounter{linenumincrement}{5}
567 %

```

`\c@firstsublinenum`    The following parameters are just like `firstlinenum` and `linenumincrement`, but for sub-line numbers. `sublinenumincrement` must be at least 1.

```

568 \newcounter{firstsublinenum}
569     \setcounter{firstsublinenum}{5}
570 \newcounter{sublinenumincrement}
571     \setcounter{sublinenumincrement}{5}
572
573 %

```

`\firstlinenum`    These macros can be used to set the corresponding counters.  
`\linenumincrement`  
`\firstsublinenum`  
`\sublinenumincrement`

```

574
575 \newcommand*{\firstlinenum}[1]{%
576     \ifledRcol%
577         \setcounter{firstlinenumR}{#1}%
578         \led@warn@setting@in@rightside{\firstlinenum}%
579     \else%
580         \setcounter{firstlinenum}{#1}%

```

```

581 \fi%
582 }
583 \newcommand*\linenumincrement}[1]{%
584 \ifledRcol%
585 \setcounter{linenumincrementR}{#1}%
586 \led@warn@setting@in@rightside{\linenumincrement}%
587 \else%
588 \setcounter{linenumincrement}{#1}%
589 \fi%
590 }
591 \newcommand*\firstsublinenum}[1]{%
592 \ifledRcol%
593 \setcounter{firstsublinenumR}{#1}%
594 \led@warn@setting@in@rightside{\firstsublinenum}%
595 \else%
596 \setcounter{firstsublinenum}{#1}%
597 \fi%
598 }
599 \newcommand*\sublinenumincrement}[1]{%
600 \ifledRcol%
601 \setcounter{sublinenumincrementR}{#1}%
602 \led@warn@setting@in@rightside{\sublinenumincrement}%
603 \else%
604 \setcounter{sublinenumincrement}{#1}%
605 \fi%
606 }
607
608 %

```

## V.4 Line number locking

`\lockdisp` When line locking is being used, the `\lockdisp{⟨word⟩}` macro specifies whether a line number—if one is due to appear—should be printed on the first printed line or on the last, or by all of them. Its argument is a word, either `first`, `last`, or `all`. Initially, it is set to `first`.

`\lock@disp` encodes the selection: 0 for first, 1 for last, 2 for all.

```

609 \newcount\lock@disp
610 \newcommand*\lockdisp}[1]{%
611 \l@dgetlock@disp{#1}%
612 \ifnum\@l@tempcntb>\m@ne
613 \global\lock@disp=\@l@tempcntb
614 \else
615 \led@warn@BadLockdisp
616 \fi}}
617 \newcommand*\l@dgetlock@disp}[1]{
618 \def\@tempa{#1}\def\@tempb{first}%
619 \ifx\@tempa\@tempb
620 \@l@tempcntb \z@

```

```

621 \else
622   \def\@tempb{last}%
623   \ifx\@tempa\@tempb
624     \@l@tempcntb \@ne
625   \else
626     \def\@tempb{all}%
627     \ifx\@tempa\@tempb
628       \@l@tempcntb \tw@
629     \else
630       \@l@tempcntb \m@ne
631     \fi
632   \fi
633 \fi}
634
635 %

```

`\sublockdisp` The same questions about where to print the line number apply to sub-lines, and these are the analogous macros for dealing with the problem.

```

636 \newcount\sublock@disp
637 \newcommand{\sublockdisp}[1]{%
638   \l@getlock@disp{#1}%
639   \ifnum\@l@tempcntb>\m@ne
640     \global\sublock@disp=\@l@tempcntb
641   \else
642     \led@warn@BadSublockdisp
643   \fi}}
644
645 %

```

## V.5 Line number style

`\linenumberstyle` We provide a mechanism for using different representations of the line numbers, not just the normal arabic.

`\linenumrep` NOTE: In v0.7 `\linenumrep` and `\sublinenumrep` replaced the internal `\linenumr@p`

`\linenumr@p` and `\sublinenumr@p`.

`\sublinenumberstyle` `\linenumberstyle` and `\sublinenumberstyle` are user level macros for setting the number representation (`\linenumrep` and `\sublinenumrep`) for line and sub-line numbers.

```

646 \newcommand*{\linenumberstyle}[1]{%
647   \def\linenumrep##1{\@nameuse{#1}{##1}}
648 \newcommand*{\sublinenumberstyle}[1]{%
649   \def\sublinenumrep##1{\@nameuse{#1}{##1}}
650 %

```

Initialise the number styles to arabic.

```

651 \linenumberstyle{arabic}
652 \let\linenumr@p\linenumrep

```

```

653 \sublinenumberstyle{arabic}
654 \let\sublinenumr@p\sublinenumrep
655
656 %

```

## V.6 Line number printing

`\leftlinenum` and `\rightlinenum` are the macros that are called to print marginal line numbers on a page, for left- and right-hand margins respectively. They are made easy to access and change, since you may want to change the styling in some way. These standard versions illustrate the general sort of thing that will be needed; they are based on the `\leftheadline` macro in *The TeXbook*, p. 416.

Whatever these macros output gets printed in a box that will be put into the appropriate margin without any space between it and the line of text. You will generally want a kern between a line number and the text, and `\linenumsep` is provided as a standard way of storing its size. Line numbers are usually printed in a smaller font, and `\numlabfont` is provided as a standard name for that font. When called, these macros will be executed within a group, so font changes and the like will remain local.

`\ledlinenum` typesets the line (and subtitle) number.

The original `\numlabfont` specification is equivalent to the  $\LaTeX$  `\scriptsize` for a 10pt document.

```

657 \newlength{\linenumsep}
658 \setlength{\linenumsep}{1pc}
659 \newcommand*{\numlabfont}{\normalfont\scriptsize}
660 \newcommand*{\ledlinenum}{%
661 \bgroup%
662 \ifluatex%
663 \textdir TL%
664 \fi%
665 \numlabfont\linenumrep{\line@num}%
666 \ifsublines@
667 \ifnum\subline@num>0\relax
668 \unskip%
669 \Xsublinesep@side%
670 \sublinenumrep{\subline@num}%
671 \fi
672 \fi%
673 \egroup%
674 }%
675
676 \newcommand*{\leftlinenum}{%
677 \ledlinenum
678 \kern\linenumsep}
679 \newcommand*{\rightlinenum}{%
680 \kern\linenumsep
681 \ledlinenum}
682

```

```
683 %
```

## V.7 Line number counters and lists

Footnote references using line numbers rather than symbols can't be generated in one pass, because we do not know the line numbers till we ship out the pages. It would be possible if footnotes were never keyed to more than one line; but some footnotes gloss passages that may run for several lines, and they must be tied to the first line of the passage glossed. And even one-line passages require two passes if we want line-per-page numbering rather than line-per-section numbering.

So we run  $\LaTeX$  over the text several times, and each time save information about page and line numbers in a 'line-list file' to be used during the next pass. At the start of each section—whenever `\beginnumbering` is executed—the line-list file for that section is read, and the information from it is encoded into a few list macros.

We need first to define the different line numbers that are involved in these macros, and the associated counters.

`\line@num` The count `\line@num` stores the line number that is used in marginal line numbering and in notes: counting either by section, page or pstart, depending on your choice for this section. This may be qualified by `\subline@num`.

```
684 \newcount\line@num
685 %
```

`\subline@num` The count `\subline@num` stores a sub-line number that qualifies `\line@num`. For example, line 10 might have sub-line numbers 1, 2 and 3, which might be printed as lines 10.1, 10.2, 10.3.

```
686 \newcount\subline@num
687 %
```

`\ifsublines@` We maintain an associated flag, `\ifsublines@`, to tell us whether we're within a sub-line range or not.  
`\sublines@true`  
`\sublines@false`

You may wonder why we do not just use the value of `\subline@num` to determine this—treating anything greater than 0 as an indication that sub-lineation is on. We need a separate flag because sub-lineation can be used together with line-number locking in odd ways: several pieces of a logical line might be interrupted by pieces of sub-lineated text, and those sub-line numbers should not return to zero until the next change in the major line number. This is common in the typesetting of English Renaissance verse drama, in which stage directions are given sub-line numbers: a single line of verse may be interrupted by several stage directions.

```
688 \newif\ifsublines@
689 %
```

`\absline@num` The count `\absline@num` stores the absolute number of lines since the start of the section: that is, the number we have actually printed, no matter what numbers we attached

to them. This value is never printed on an output page, though `\line@num` will often be equal to it. It is used internally to keep track of where notes are to appear and where new pages start: using this value rather than `\line@num` is a lot simpler, because it does not depend on the lineation system in use.

```
690 \newcount\absline@num
691 %
```

We will call `\absline@num` numbers “absolute” numbers, and `\line@num` and `\subline@num` numbers “visible” numbers.

## V.8 Line number locking counter

`\@lock`    The counts `\@lock` and `\sub@lock` tell us the state of line-number and sub-line-number locking. 0 means we are not within a locked set of lines; 1 means we are at the first line in the set; 2, at some intermediate line; and 3, at the last line.

```
692 \newcount\@lock
693 \newcount\sub@lock
694 %
```

## V.9 Line number associated to lemma

`\line@list`    Now we can define the list macros that will be created from the line-list file. We will maintain the following lists:

`\insertlines@list`  
`\actionlines@list`  
`\actions@list`

- `\line@list`: the page and line numbers for every lemma marked by `\edtext`. There are seven pieces of information, separated by vertical bars:

1. the starting page,
2. line, and
3. sub-line numbers, followed by the
4. ending page,
5. line, and
6. sub-line numbers, and then the
7. font specifier for the lemma.

These line numbers are all visible numbers. The font specifier is a set of four codes for font encoding, family, series, and shape, separated by / characters. Thus a lemma that started on page 23, line 35 and went on until page 24, line 3 (with no sub-line numbering), and was typeset in a normal roman font would have a line list entry like this:

```
23|35|0|24|3|0|OT1/cm/r/m/n.
```

There is one item in this list for every lemma marked by `\edtext`, even if there are several notes to that lemma, or no notes at all. `\edtext` reads the data in this list, making it available for use in the text of notes.

- `\insertlines@list`: the line numbers of lines that have footnotes or other insertions. These are the absolute numbers where the corresponding lemmas begin. This list contains one entry for every footnote in the section; one lemma may contribute no footnotes or many footnotes. This list is used by `\add@inserts` within `\do@line`, to tell it where to insert notes.
- `\actionlines@list`: a list of absolute line numbers at which we are to perform special actions; these actions are specified by the `\actions@list` list defined below.
- `\actions@list`: action codes corresponding to the line numbers in `\actionlines@list`. These codes tell `reledmac` what action it is supposed to take at each of these lines. One action, the page-start action, is generated behind the scenes by `reledmac` itself; the others, for specifying sub-lineation, line-number locking, and line-number alteration, are generated only by explicit commands in your input file. The page-start and line-number-alteration actions require arguments, to specify the new values for the page or line numbers; instead of storing those arguments in another list, we have chosen the action-code values so that they can encode both the action and the argument in these cases. Action codes greater than  $-1000$  are page-start actions, and the code value is the page number; action codes less than  $-5000$  specify line numbers, and the code value is a transformed version of the line number; action codes between these two values specify other actions which require no argument.

Here is the full list of action codes and their meanings:

Any number greater than  $-1000$  is a page-start action: the line number associated with it is the first line on a page, and the action number is the page number. (The cutoff of  $-1000$  is chosen because negative page-number values are used by some macro packages; we assume that page-number values less than  $-1000$  are not common.) Page-start action codes are added to the list by the `\page@action` macro, which is (indirectly) triggered by the workings of the `\page@start` macro; that macro should always be called in the output routine, just before the page contents are assembled. `Eledmac` calls it in `\pagecontents`.

The action code  $-1001$  specifies the start of sub-lineation: meaning that, starting with the next line, we should be advancing `\subline@num` at each start-of-line command, rather than `\line@num`.

The action code  $-1002$  specifies the end of sub-lineation. At the next start-of-line, we should clear the sub-line counter and start advancing the line number. The action codes for starting and ending sub-lineation are added to the list by the `\sub@action` macro, as called to implement the `\startsub` and `\endsub` macros.

The action code  $-1003$  specifies the start of line number locking. After the number for the current line is computed, it will remain at that value through the next line that has an action code to end locking.

The action code  $-1004$  specifies the end of line number locking.



The action code `-1005` specifies the start of sub-line number locking. After the number for the current sub-line is computed, it will remain at that value through the next sub-line that has an action code to end locking.

The action code `-1006` specifies the end of sub-line number locking.

The four action codes for line and sub-line number locking are added to the list by the `\do@lockon` and `\do@lockoff` macros, as called to implement the `\startlock` and `\endlock` macros.

An action code of `-5000` or less sets the current visible line number (either the line number or the sub-line number, whichever is currently being advanced) to a specific positive value. The value of the code is  $-(5000 + n)$ , where  $n$  is the value (always  $\geq 0$ ) assigned to the current line number. Action codes of this type are added to the list by the `\set@line@action` macro, as called to implement the `\advanceline` and `\setline` macros: this action only occurs when the user has specified some change to the line numbers using those macros. Normally `reledmac` computes the visible line numbers from the absolute line numbers with reference to the other action codes and the settings they invoke; it does not require an entry in the action-code list for every line.

Here are the commands to create these lists:

```
695 \list@create{\line@list}
696 \list@create{\insertlines@list}
697 \list@create{\actionlines@list}
698 \list@create{\actions@list}
699
700 %
```

`\page@num` We will need some counts while we read the line-list, for the page number and the ending  
`\endpage@num` page, line, and sub-line numbers. Some of these will be used again later on, when we  
`\endline@num` are acting on the data in our list macros.  
`\endsubline@num`

```
701 \newcount\page@num
702 \newcount\endpage@num
703 \newcount\endline@num
704 \newcount\endsubline@num
705 %
```

`\ifnoteschanged@` If the number of the footnotes in a section is different from what it was during the last  
`\noteschanged@true` run, or if this is the very first time you've run  $\text{\LaTeX}$ , on this file, the information from  
`\noteschanged@false` the line-list used to place the notes will be wrong, and some notes will probably be  
misplaced. When this happens, we prefer to give a single error message for the whole  
section rather than messages at every point where we notice the problem, because we do  
not really know where in the section notes were added or removed, and the solution in  
any case is simply to run  $\text{\LaTeX}$  two more times; there is no fix needed to the document.  
The `\ifnoteschanged@` flag is set if such a change in the number of notes is discovered  
at any point.

```
706 \newif\ifnoteschanged@
707 %
```

**\resetprevline@** Inside the apparatus, at each note, the line number is stored in a macro called `\prevlineX`, where X is the letter of the current series. This macro is called when using `\Xnumberonlyfirstinline`. This macro must be reset at the same time as the line number. The `\resetprevline@` does this resetting for every series.

```
\resetprevline@08 \newcommand*\resetprevline@{%
709   \def\do##1{\global\csundef{prevline##1}}%
710   \dolistloop{\@series}%
711 }
712 %
```

**\resetprevpage@num** Inside the apparatus, at each note, the page number is stored in a macro called `\prevpageX@num`, where X is the letter of the current series. This macro is called when using `\Xparafootsep` or `\parafootsepX`. This macro must be reset at the beginning of each numbered section. The `\resetprevpage@` command resets this macro for every series.

```
\resetprevpage@13 \newcommand*\resetprevpage@num{%
714   \def\do##1{\ifcsdef{prevpage##1@num}{\global\csname prevpage##1@num\
endcsname=0}{}}%
715   \dolistloop{\@series}%
716 }
717 %
```

## V.10 Reading the line-list file

**\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. First, it clear all previous line's list.

```
718 \newread\@inputcheck
719 \newcommand*\read@linelist}[1]{%
720   \ifledRcol%
721   \list@clearing@regR%
722   \else%
723   \list@clearing@reg%
724   \fi%
725 %
```

When using `reledpar`, make sure that the `\maxlinesinpar@list` is empty (otherwise things will be thrown out of kilter if there is any old stuff still hanging in there).

```
726   \list@clear{\maxlinesinpar@list}
727 %
```

Now get the file and interpret it. When the file is there we start a new group and make some special definitions we will need to process it. It is a sequence of  $\TeX$  commands, but they require a few special settings. We make [ and ] become grouping characters: they are used that way in the line-list file, because we need to write them out one at a time rather than in balanced pairs, and it is easier to just use something other than real braces. @ must become a letter, since this is run in the ordinary  $\LaTeX$  context. We ignore carriage returns, since if we are in horizontal mode they can get interpreted as spaces to be printed.

Our line, page, and line-locking counters were already zeroed by `\line@list@stuff` if this is being called from within `\beginnumbering`; sub-lineation will be turned off as well in that case. On the other hand, if this is being called from `\resumenumbers`, those things should still have the values they had when `\pausenumbering` was executed.

If the file is not there, we print an informative message.

Now, after these preliminaries, we start interpreting the file.

```
728 \get@linelistfile{#1}%
729 \endgroup
730 %
```

When the reading is done, we are all through with the line-list file. All the information we needed from it will now be encoded in our list macros.

Finally, we initialize the `\next@actionline` and `\next@action` macros, which specify where and what the next action to be taken is.

```
731 \ifledRcol
732   \global\page@numR=\m@ne
733   \ifx\actionlines@listR\empty
734     \gdef\next@actionlineR{1000000}%
735   \else
736     \gl@p\actionlines@listR\to\next@actionlineR
737     \gl@p\actions@listR\to\next@actionR
738   \fi
739 \else
740   \global\page@num=\m@ne
741   \ifx\actionlines@list\empty
742     \gdef\next@actionline{1000000}%
743   \else
744     \gl@p\actionlines@list\to\next@actionline
745     \gl@p\actions@list\to\next@action
746   \fi
747 \fi
748 }
749 %
```

`\list@clearing@reg` Clears the lists for `\read@linelist`

```
750 \newcommand*{\list@clearing@reg}{%
751   \list@clear{\line@list}%
752   \list@clear{\insertlines@list}%
```

```

753 \list@clear{\actionlines@list}%
754 \list@clear{\actions@list}%
755 \list@clear{\linesinpar@listL}%
756 \list@clear{\linesonpage@listL}%
757 }%
758 %

```

`\get@linelistfile` reledmac can take advantage of the  $\LaTeX$  ‘safe file input’ macros to get the line-list file.

```

759 \newcommand*\get@linelistfile}[1]{%
760   \InputIfFileExists{#1}{%
761     \global\noteschanged@false
762     \begingroup
763       \catcode`\[=1 \catcode`\]=2
764       \makeatletter \catcode`\^M=9}{%
765     \led@warn@NoLineFile{#1}%
766     \global\noteschanged@true
767     \begingroup}%
768   }
769 %
770 %

```

This version of `\read@linelist` creates list macros containing data for the entire section, so they could get rather large. It would be no more difficult to read the line-list file incrementally rather than all at once: we could read, at the start of each paragraph, only the commands relating to that paragraph. But this would require that we have two line-lists open at once, one for reading, one for writing, and on systems without version numbers we would have to do some file renaming outside of  $\LaTeX$  for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the `\pausenumbering` and `\resumenumbering` macros to help you if you run into macro memory limitations (see 4.2.7 p. 18 above).

## V.11 Commands within the line-list file

This section defines the commands that can appear within a line-list file. They all have very short names because we are likely to be writing very large numbers of them out. One macro, `\@nl`, is especially short, since it will be written to the line-list file once for every line of text in a numbered section. (Another of these commands, `\@lab`, will be introduced in a later section, among the cross-referencing commands it is associated with.)

When these commands modify the various page and line counters, they deliberately do not use `\global`. This is because we want them to affect only the counter values within the current group when nested calls of `\@ref` occur. (The code assumes throughout that the value of `\globaldefs` is zero.)

The macros with `action` in their names contain all the code that modifies the action-code list: again, this is so that they can be turned off easily for nested calls of `\@ref`.

`\line@list@version` The `\line@list@version` check if the line-list file does not refers to the older commands of `reledmac`. In this case, we stop reading the line-list file. Consequently, `\line@list@version` must be the first line of a line-number file.

```

771 \newcommand{\line@list@version}[1]{%
772   \IfStrEq{#1}{\this@line@list@version}%
773   }%
774   {\ifledRcol%
775     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
776     \else%
777     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
778     \fi%
779   \endinput%
780   }%
781 }%
782 %

```

`\@nl` `\@nl` does everything related to the start of a new line of numbered text.

`\@nl@reg` In order to get the `\setlinenum` to work Peter Wilson had to slip in some new code at the start of the macro, to get the timing of the actions correct. The problem was that his original naive implementation of `\setlinenum` had a unfortunate tendency to change the number of the last line of the *preceding* paragraph. The new code is sort of based on the page number handling and `\setline`. It seems that a lot of fiddling with the line number internals is required.

In November 2004 in order to accurately determine page numbers Peter Wilson added these to the macro. It is now:

```
\@nl{<page counter number>}{<printed page number>}
```

We do not (yet) use the printed number (i.e., the `\thepage`) but it may come in handy later. The macro `\fix@page` checks if a new page has started.

Exactly what `\@nl` does depends on whether right text is being processed. That's why many code is defined in `\@nl@reg` or `\nl@regR`.

```

783
784 \newcommand*{\@nl}[2]{%
785   \fix@page{#1}%
786   \ifledRcol%
787     \@nl@regR%
788   \else%
789     \@nl@reg%
790   \fi%
791 }
792 \newcommand*{\@nl@reg}{%
793   \ifx\l@dchset@num\relax \else
794     \advance\absline@num \@ne
795     \set@line@action
796     \let\l@dchset@num=\relax
797     \advance\absline@num \m@ne
798     \advance\line@num \m@ne
799   \fi

```

```

800 %
      First increment the absolute line-number, and perform deferred actions relating to
      page starts and sub-lines.
801 \advance\absline@num \@ne
      \ifx\next@page@num\relax \else
802     \page@action
803     \let\next@page@num=\relax
804 \fi
805 \ifx\sub@change\relax \else
806     \ifnum\sub@change>\z@
807         \sublines@true
808     \else
809         \sublines@false
810     \fi
811     \sub@action
812     \let\sub@change=\relax
813 \fi
814 %
815 %

```

Fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

816 \ifcase\@lock
817     \or
818     \@lock \tw@
819     \or \or
820     \@lock \z@
821 \fi
822 \ifcase\sub@lock
823     \or
824     \sub@lock \tw@
825     \or \or
826     \sub@lock \z@
827 \fi
828 %

```

Now advance the visible line number, unless it has been locked.

```

829 \ifsublines@
830     \ifnum\sub@lock<\tw@
831         \advance\subline@num \@ne
832     \fi
833 \else
834     \ifnum\@lock<\tw@
835         \advance\line@num \@ne \subline@num \z@
836     \fi
837 \fi}
838 %
839 %

```

`\last@page@num` `\fix@page` basically replaces `\@page`. It determines whether or not a new page has been started, based on the page values held by `\@n1`.

```

840 \newcount\last@page@num
841   \last@page@num=-10000
842
843 \newcommand*\fix@page}[1]{%
844   \ifledRcol
845     \ifnum #1=\last@page@numR
846       \else
847         \ifbypage@R
848           \line@numR \z@ \subline@numR \z@
849         \fi
850         \global\page@numR=#1\relax
851         \global\last@page@numR=#1\relax
852         \def\next@page@numR{#1}%
853       \fi
854     \else
855       \ifnum #1=\last@page@num
856         \else
857           \ifbypage@
858             \line@num \z@ \subline@num \z@
859           \fi
860           \global\page@num=#1\relax
861           \global\last@page@num=#1\relax
862           \def\next@page@num{#1}%
863           \listxadd{\normal@page@break}{\the\absline@num}
864         \fi
865       \fi}
866 %

```

`\@pend` These do not do anything at this point, but will have been added to the auxiliary file(s) if the `reledpar` package has been used. They are just here to stop `reledmac` from moaning if the `reledpar` is used for one run and then not for the following one.

```

\@lopL
\@lopR
867 \newcommand*\@pend}[1]{}
868 \newcommand*\@pendR}[1]{}
869 \newcommand*\@lopL}[1]{}
870 \newcommand*\@lopR}[1]{}
871
872 %

```

`\sub@on` The `\sub@on` and `\sub@off` macros turn sub-lineation on and off: but not directly, since such changes do not really take effect until the next line of text. Instead they set a flag that notifies `\@n1` of the necessary action.

```

873 \newcommand*\sub@on{\ifsublines@
874   \let\sub@change=\relax
875   \else
876     \def\sub@change{1}%

```

```

877 \fi}
878 \newcommand*{\sub@off}{\ifsublines@
879   \def\sub@change{-1}%
880   \else
881   \let\sub@change=\relax
882 \fi}
883
884 %

```

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

```

885
886 \newcommand*{\@adv}[1]{%
887   \ifsublines@
888     \ifledRcol
889       \advance\subline@numR by #1\relax
890       \ifnum\subline@numR<\z@
891         \led@warn@BadAdvancelineSubline
892         \subline@numR \z@
893       \fi
894     \else
895       \advance\subline@num by #1\relax
896       \ifnum\subline@num<\z@
897         \led@warn@BadAdvancelineSubline
898         \subline@num \z@
899       \fi
900     \fi
901   \else
902     \ifledRcol
903       \advance\line@numR by #1\relax
904       \ifnum\line@numR<\z@
905         \led@warn@BadAdvancelineLine
906         \line@numR \z@
907       \fi
908     \else
909       \advance\line@num by #1\relax
910       \ifnum\line@num<\z@
911         \led@warn@BadAdvancelineLine
912         \line@num \z@
913       \fi
914     \fi
915   \fi
916   \set@line@action}
917
918 %

```

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



```

919
920 \newcommand*{\@set}[1]{%
921   \ifledRcol
922     \ifsublines@
923       \subline@numR=#1\relax
924     \else
925       \line@numR=#1\relax
926     \fi
927   \set@line@action
928 \else
929   \ifsublines@
930     \subline@num=#1\relax
931   \else
932     \line@num=#1\relax
933   \fi
934   \set@line@action
935 \fi}
936
937 %

```

`\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`.

`\l@dchset@num`

`\l@dchset@num` is a flag to the `\@nl?` macro. If it is not `\relax` then a linenum change is to be done.

```

938
939 \newcommand*{\l@d@set}[1]{%
940   \ifledRcol
941     \line@numR=#1\relax
942     \advance\line@numR \@ne
943     \def\l@dchset@num{#1}
944   \else
945     \line@num=#1\relax
946     \advance\line@num \@ne
947     \def\l@dchset@num{#1}
948   \fi}
949 \let\l@dchset@num\relax
950
951 %

```

`\page@action` `\page@action` adds an entry to the action-code list to change the page number.

```

952
953 \newcommand*{\page@action}{%
954   \ifledRcol
955     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
956     \xright@appenditem{\next@page@numR}\to\actions@listR
957   \else
958     \xright@appenditem{\the\absline@num}\to\actionlines@list
959     \xright@appenditem{\next@page@num}\to\actions@list

```

```
960 \fi}
961 %
```

**\set@line@action** \set@line@action adds an entry to the action-code list to change the visible line number.

```
962
963 \newcommand*\set@line@action){%
964   \ifledRcol
965     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
966     \ifsublines@
967       \@l@tempcnta=-\subline@numR
968     \else
969       \@l@tempcnta=-\line@numR
970     \fi
971     \advance\@l@tempcnta by -5000\relax
972     \xright@appenditem{\the\@l@tempcnta}\to\actions@listR
973   \else
974     \xright@appenditem{\the\absline@num}\to\actionlines@list
975     \ifsublines@
976       \@l@tempcnta=-\subline@num
977     \else
978       \@l@tempcnta=-\line@num
979     \fi
980     \advance\@l@tempcnta by -5000\relax
981     \xright@appenditem{\the\@l@tempcnta}\to\actions@list
982   \fi}
983 %
```

**\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.

```
984
985 \newcommand*\sub@action){%
986   \ifledRcol
987     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
988     \ifsublines@
989       \xright@appenditem{-1001}\to\actions@listR
990     \else
991       \xright@appenditem{-1002}\to\actions@listR
992     \fi
993   \else
994     \xright@appenditem{\the\absline@num}\to\actionlines@list
995     \ifsublines@
996       \xright@appenditem{-1001}\to\actions@list
997     \else
998       \xright@appenditem{-1002}\to\actions@list
999     \fi
1000   \fi}
1001 %
```

`\lock@on` `\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.

Adding commands to the action list is slow, and it is very often the case that a lock-on command is immediately followed by a lock-off command in the line-list file, and therefore really does nothing. We use a look-ahead scheme here to detect such pairs, and add nothing to the line-list in those cases.

```

1002 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
1003
1004 \newcommand*{\do@lockon}{%
1005   \ifx\next\lock@off
1006     \global\let\lock@off=\skip@lockoff
1007   \else
1008     \ifledRcol
1009       \do@lockonR
1010     \else
1011       \do@lockonL
1012     \fi
1013   \fi}
1014
1015
1016 \newcommand*{\do@lockonL}{%
1017   \xright@appenditem{the\absline@num}\to\actionlines@list
1018   \ifsublines@
1019     \xright@appenditem{-1005}\to\actions@list
1020     \ifnum\sub@lock=\z@
1021       \sub@lock \@ne
1022     \else
1023       \ifnum\sub@lock=\thr@@
1024         \sub@lock \@ne
1025       \fi
1026     \fi
1027   \else
1028     \xright@appenditem{-1003}\to\actions@list
1029     \ifnum\@lock=\z@
1030       \@lock \@ne
1031     \else
1032       \ifnum\@lock=\thr@@
1033         \@lock \@ne
1034       \fi
1035     \fi
1036   \fi}
1037
1038 %

```

`\lock@off` `\lock@off` adds an entry to the action-code list to turn line number locking off.

```

\do@lockoff
\do@lockoffL1039 \newcommand*{\do@lockoffL}{%
\skip@lockoff1040 \xright@appenditem{the\absline@num}\to\actionlines@list

```

```

1041 \ifsublines@
1042 \xright@appenditem{-1006}\to\actions@list
1043 \ifnum\sub@lock=\tw@
1044 \sub@lock \thr@@
1045 \else
1046 \sub@lock \z@
1047 \fi
1048 \else
1049 \xright@appenditem{-1004}\to\actions@list
1050 \ifnum\@lock=\tw@
1051 \@lock \thr@@
1052 \else
1053 \@lock \z@
1054 \fi
1055 \fi}
1056
1057 \newcommand*\do@lockoff}{%
1058 \ifledRcol
1059 \do@lockoffR
1060 \else
1061 \do@lockoffL
1062 \fi}
1063 \newcommand*\skip@lockoff}{\global\let\lock@off=\do@lockoff}
1064 \global\let\lock@off=\do@lockoff
1065
1066 %

```

`\n@num` These macros implement the `\skipnumbering` command. They use action code 1007.

```

1067 \newcommand*\n@num}{%
1068 \ifledRcol%
1069 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1070 \xright@appenditem{-1007}\to\actions@listR
1071 \else%
1072 \xright@appenditem{\the\absline@num}\to\actionlines@list%
1073 \xright@appenditem{-1007}\to\actions@list%
1074 \fi%
1075 }%
1076
1077 %

```

`\n@num@stanza` This macro implements the `\skipnumbering` for stanza command. It uses action code 1008.

```

1078 \newcommand*\n@num@stanza}{%
1079 \ifledRcol%
1080 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1081 \xright@appenditem{-1008}\to\actions@listR%
1082 \else%

```

```

1083 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1084 \xright@appenditem{-1008}\to\actions@list%
1085 \fi%
1086 }
1087 %

```

`\ifl@dhidnumber` `\hidnumbering` hides number in margin. It uses action code 1009.  
`\hidnumbering`

```

\h@num88 \newif\ifl@dhidnumber
1089 \newcommand*{\hidnumbering}{
1090 \ifledRcol%
1091 \write\linenum@outR{\string\hide@num}%
1092 \else%
1093 \write\linenum@out{\string\hide@num}%
1094 \fi%
1095 }%
1096 \newcommand*{\hide@num}{%
1097 \ifledRcol%
1098 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1099 \xright@appenditem{-1009}\to\actions@listR%
1100 \else%
1101 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1102 \xright@appenditem{-1009}\to\actions@list%
1103 \fi%
1104 }
1105 %

```

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

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

```

1106 \newcount\insert@count
1107 %

```

- #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`.)

`\dummy@ref` When nesting of `\@ref` commands does occur, it is necessary to temporarily redefine `\@ref` within `\@ref`, so that we are only doing one of these at a time.

```

1108 \newcommand*{\dummy@ref}[2]{#2}
1109 %

```

`\@ref@reg` The first thing `\@ref` (i.e. `\@ref@reg`) itself does is to add the specified number of items to the `\insertlines@list` list.

```

1110 \newcommand*\@ref}[2]{%
1111   \ifledRcol%
1112     \@ref@regR{#1}{#2}%
1113   \else%
1114     \@ref@reg{#1}{#2}%
1115   \fi%
1116 }%
1117 \newcommand*\@ref@reg}[2]{%
1118   \global\insert@count=#1\relax
1119   \global\advance\@edtext@level by 1%
1120   \loop\ifnum\insert@count>\z@
1121     \xright@appenditem{\the\absline@num}\to\insertlines@list
1122     \global\advance\insert@count \m@ne
1123   \repeat
1124 %

```

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.

```

1125 \begingroup
1126   \let\@ref=\dummy@ref
1127   \let\@lopL@\gobble
1128   \let\page@action=\relax
1129   \let\sub@action=\relax
1130   \let\set@line@action=\relax
1131   \let\@lab=\relax
1132   \let\@lemma=\relax%
1133   \let\@sw@\gobblethree%
1134   #2
1135   \global\endpage@num=\page@num
1136   \global\endline@num=\line@num
1137   \global\endsubline@num=\subline@num
1138 \endgroup
1139 %

```

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

```

1140 \xright@appenditem%
1141   {\the\page@num|\the\line@num|%
1142     \ifsublines@ \the\subline@num \else 0\fi|%
1143     \the\endpage@num|\the\endline@num|%
1144     \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1145 %

```

Create a list which stores every second argument of each `\@sw` in this lemma, at this level. Also set the boolean about the use of lemma in this edtext level to false.

```

1146 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
@edtext@level\endcsname}%

```

```

1147 \providebool{lemmacommand@the\edtext@level}%
1148 \boolfalse{lemmacommand@the\edtext@level}%
1149 %

```

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

```

1150 #2%
1151 %

```

Now, we store the list of `\@sw` of this current `\edtext` as an element of the global list of list of `\@sw` for a `\edtext` depth.

```

1152 \ifnum\edtext@level>0%
1153 \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
csname sw@list@edtext@the\edtext@level\endcsname}}%
1154 \ifcsundef{sw@list@edtext@the\edtext@level}{\create@this@edtext@level
}{}%
1155 \letcs{\@tmp}{sw@list@edtext@the\edtext@level}%
1156 \letcs{\@tmpp}{sw@list@edtext@tmp@the\edtext@level}
1157 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
1158 \global\cslet{sw@list@edtext@the\edtext@level}{\@tmp}%
1159 \fi%
1160 %

```

Decrease `edtext` level counter.

```

1161 \global\advance\edtext@level by -1%
1162 %
1163 }
1164
1165 %

```

## V.12 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@out` The file will be opened on output stream `\linenum@out`.

```

1166 \newwrite\linenum@out
1167 %

```

`\iffirst@linenum@out@` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open. The reason is that we want the output routine to write the page number for every page to this file; otherwise we would have to write it at the start of every line. But it is not very easy for the output routine to tell whether an output stream is open or not. There is no way to test the status of a particular output stream directly,

and the asynchronous nature of output routines makes the status hard to determine by other means.

We can manage pretty well by means of the `\iffirst@linenum@out@` flag; its inelegant name suggests the nature of the problem that made its creation necessary. It is set to be true before any `\linenum@out` file is opened. When such a file is opened for the first time, it is done using `\immediate`, so that it will at once be safe for the output routine to write to it; we then set this flag to false.

```
1168 \newif\iffirst@linenum@out@
1169   \first@linenum@out@true
1170 %
```

`\this@line@list@version` The commands allowed in the line-list file and their arguments can change between two versions of `reledmac`. The `\this@line@list@version` command is upgraded when it happens. It is written in the file list. If we process a line-list file which used an older version, that means the commands used inside are deprecated, and we can't use them.

```
1171 \newcommand{\this@line@list@version}{5}%
1172 %
```

`\line@list@stuff` The `\line@list@stuff{<file>}` macro, which is called by `\beginnumbering`, performs all the line-list operations needed at the start of a section. Its argument is the name of the line-list file.

```
1173 \newcommand*{\line@list@stuff}[1]{%
1174 %
```

First, use the commands of the previous section to interpret the line-list file from the last run.

```
1175   \read@linelist{#1}%
1176 %
```

Now close the current output line-list file, if any, and open a new one. The first time we open a line-list file for output, we do it using `\immediate`, and clear the `\iffirst@linenum@out@` flag.

```
1177   \iffirst@linenum@out@
1178     \immediate\closeout\linenum@out%
1179     \global\first@linenum@out@false%
1180     \immediate\openout\linenum@out=#1\relax%
1181     \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}%
1182     \ifl@dpaging%
1183       \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}%
1184     \fi%
1185   \else
1186 %
```

If we get here, then this is not the first line-list we have seen, so we do not open or close the files immediately.



```

1187 \if@minipage%
1188 \leavevmode%
1189 \fi%
1190 \closeout\linenum@out%
1191 \openout\linenum@out=#1\relax%
1192 \write\linenum@out{\string\line@list@version{\this@line@list@version}}
%
1193 \ifl@dpaging%
1194 \write\linenum@out{\string\@par@sync@option{\@par@this@sync@option}}
%
1195 \fi%
1196 \fi}
1197
1198 %

```

`\new@line` The `\new@line` macro sends the `\@nl` command to the line-list file, to mark the start of a new text line, and its page number.

```

1199 \newcommand*{\new@line}{%
1200 \IfStrEq{\led@pb@setting}{after}%
1201 {\xifinlist{the\absline@num}{\l@prev@nopb}%
1202 {\xifinlist{the\absline@num}{\normal@page@break}%
1203 {\numgdef{\@next@page}{\c@page+1}%
1204 \write\linenum@out{\string\@nl[\@next@page][\@next@page]}}%
1205 }%
1206 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1207 }%
1208 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1209 }%
1210 \IfStrEq{\led@pb@setting}{before}%
1211 {\numdef{\next@absline}{\the\absline@num+1}%
1212 \xifinlist{\next@absline}{\l@prev@nopb}%
1213 {\xifinlist{the\absline@num}{\normal@page@break}%
1214 {\numgdef{\nc@page}{\c@page+1}%
1215 \write\linenum@out{\string\@nl[\nc@page][\nc@page]}}%
1216 }%
1217 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1218 }%
1219 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1220 }%
1221 {}%
1222 \IfStrEqCase{\led@pb@setting}{before}{\relax}{after}{\relax}}{\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1223 }
1224
1225 %

```

`\if@noneed@Footnote` `\if@noneed@Footnote` is a boolean to check if we have to print a error message when a `\edtext` is called without any critical notes.

`\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. `\edtext` is responsible for setting the value of `\insert@count` appropriately; it actually gets done by the various footnote macros.

```

1226 \newif\if@noneed@Footnote%
1227
1228 \newcommand*{\flag@start}{%
1229   \ifledRcol%
1230     \edef\next{\write\linenum@outR{%
1231       \string\@ref[\the\insert@countR] []}}%
1232     \next%
1233     \ifnum\insert@countR<1%
1234       \if@noneed@Footnote\else%
1235         \led@err@EdtextWithoutFootnote%
1236       \fi%
1237     \fi%
1238   \else%
1239     \edef\next{\write\linenum@out{%
1240       \string\@ref[\the\insert@count] []}}%
1241     \next%
1242     \ifnum\insert@count<1%
1243       \if@noneed@Footnote\else%
1244         \led@err@EdtextWithoutFootnote%
1245       \fi%
1246     \fi%
1247   \fi}%
1248
1249 %

```

`\startsub` `\endsub` `\startsub` and `\endsub` turn sub-lineation on and off, by writing appropriate instructions to the line-list file. When sub-lineation is in effect, the line number counter is frozen and the sub-line counter advances instead. If one of these commands appears in the middle of a line, it does not take effect until the next line; in other words, a line is counted as a line or sub-line depending on what it started out as, even if that changes in the middle.

We tinker with `\lastskip` because a command of either sort really needs to be attached to the last word preceding the change, not the first word that follows the change. This is because sub-lineation will often turn on and off in mid-line—stage directions, for example, often are mixed with dialogue in that way—and when a line is mixed we want to label it using the system that was in effect at its start. But when sub-lineation begins at the very start of a line we have a problem, if we don't put in this code.

```

1250
1251
1252 \newcommand*{\startsub}{\dimen0\lastskip
1253   \ifdim\dimen0>Opt \unskip \fi
1254   \ifledRcol \write\linenum@outR{\string\sub@on}}%
1255   \else \write\linenum@out{\string\sub@on}}%
1256   \fi

```

```

1257 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1258 \def\endsub{\dimen0\lastskip
1259 \ifdim\dimen0>0pt \unskip \fi
1260 \ifledRcol \write\linenum@outR{\string\sub@off}%
1261 \else \write\linenum@out{\string\sub@off}%
1262 \fi
1263 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1264
1265 %

```

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

```

1266 \newcommand*{\advanceline}[1]{\leavevmode%
1267 \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
1268 \else \write\linenum@out{\string\@adv[#1]}%
1269 \fi%
1270 }
1271 %

```

**\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.

```

1272
1273 \newcommand*{\setline}[1]{%
1274 \leavevmode%
1275 \ifnum#1<\z@
1276 \led@warn@BadSetline
1277 \else
1278 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1279 \else \write\linenum@out{\string\@set[#1]}%
1280 \fi
1281 \fi}
1282
1283 %

```

**\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.

```

1284
1285 \newcommand*{\setlinenum}[1]{%
1286 \ifnum#1<\z@
1287 \led@warn@BadSetlinenum
1288 \else
1289 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1290 \else \write\linenum@out{\string\l@d@set[#1]} \fi
1291 \fi}
1292
1293 %

```

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

```

1294
1295 \newcommand*{\startlock}{%
1296   \ifledRcol \write\linenum@outR{\string\lock@on}%
1297   \else      \write\linenum@out{\string\lock@on}%
1298   \fi}
1299 \def\endlock{%
1300   \ifledRcol \write\linenum@outR{\string\lock@off}%
1301   \else      \write\linenum@out{\string\lock@off}%
1302   \fi}
1303 %

```

`\ifl@dskipnumber` In numbered text `\skipnumbering` will suspend the numbering for that particular line.  
`\ifl@dskipversenumber`

```

\l@dskipnumbertrue
\l@dskipnumberfalse
\skipnumbering
1304 \newif\ifl@dskipnumber
1305 \newif\ifl@dskipversenumber%
1306 \newcommand*{\skipnumbering}{%
1307   \leavevmode%
1308   \ifledRcol%
1309     \ifinstanza%
1310       \write\linenum@outR{\string\n@num@stanza}%
1311     \else%
1312       \write\linenum@outR{\string\n@num}%
1313     \fi%
1314     \advanceline{-1}%
1315   \else%
1316     \ifinstanza%
1317       \write\linenum@out{\string\n@num@stanza}%
1318     \else%
1319       \write\linenum@out{\string\n@num}%
1320     \fi%
1321     \advanceline{-1}%
1322   \fi%
1323 }%
1324
1325 %

```

## VI 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.

The `\edtext` macro takes two arguments.

`\edtext{#1}{#2}`

- #1 is the piece of the main text being glossed; it gets added to the main text, and is also used as a lemma for notes to it.
- #2 is a series of subsidiary macros that generate various kinds of notes.

The `\edtext` macro may be used (somewhat) recursively; that is, `\edtext` may be used within its own first argument. The code would be much simpler without this feature, but nested notes will commonly be necessary: it is quite likely that we will have an explanatory note for a long passage and notes on variants for individual words within that passage. The situation we can't handle is overlapping notes that are not nested: for example, one note covering lines 10–15, and another covering 12–18. You can handle such cases by using the `\lemma` and `\linenum` macros within #2: they alter the copy of the lemma and the line numbers that are passed to the notes, and hence allow you to overcome any limitations of this system, albeit with extra effort.

The recursive operation of `\edtext` will fail if you try to use a copy that is called something other than `\edtext`. In order to handle recursion, `\edtext` needs to redefine its own definition temporarily at one point, and that does not work if the macro you are calling is not actually named `\edtext`. There is no problem as long as `\edtext` is not invoked in the first argument. If you want to call `\edtext` something else, it is best to create instead a macro that expands to an invocation of `\edtext`, rather than copying `\edtext` and giving it a new name; otherwise you will need to add an appropriate definition for your new macro to `\morenoexpands`.

Side effects of our line-numbering code make it impossible to use the usual footnote macros directly within a paragraph whose lines are numbered (see comments to `\do@line`, VII.2.1 p. 129). Instead, the appropriate note-generating command is appended to the list macro `\inserts@list`, and when `\pend` completes the paragraph it inserts all the notes at the proper places.

Note that we do not provide previous-note information, although it is often wanted; your own macros must handle that. We cannot do it correctly without keeping track of what kind of notes have gone past: it is not just a matter of remembering the line numbers associated with the previous invocation of `\edtext`, because that might have been for a different kind of note. It is preferable for your footnote macros to store and recall this kind of information if they need it.

## VI.1 `\edtext` itself

The various note-generating macros might want to request that commands be executed not at once, but in close connection with the start or end of the lemma. For example, footnote numbers in the text should be connected to the end of the lemma; or, instead of a single macro to create a note listing variants, you might want to use several macros in series to create individual variants, which would each add information to a private macro or token register, which in turn would be formatted and output when all of #2 for the lemma has been read.

`\end@lemmas` To accommodate this, we provide a list macro to which macros may add commands that should subsequently be executed at the end of the lemma when that lemma is added to the text of the paragraph. A macro should add its contribution to `\end@lemmas` by using `\xleft@appenditem`. (Anything that needs to be done at the *start* of the lemma may be handled using `\aftergroup`, since the commands specified within `\edtext`'s second argument are executed within a group that ends just before the lemma is added to the main text.)

`\end@lemmas` is intended for the few things that need to be associated with the end of the lemma, like footnote numbers. Such numbers are not implemented in the current version, and indeed no use is currently made of `\end@lemmas` or of the `\aftergroup` trick. The general approach would be to define a macro to be used within the second argument of `\edtext` that would add the appropriate command to `\end@lemmas`.

Commands that are added to this list should always take care not to do anything that adds possible line-breaks to the output; otherwise line numbering could be thrown off.

```
1326 \list@create{\end@lemmas}
1327 %
```

`\dummy@edtext` We now need to define a number of macros that allow us to weed out nested instances of `\edtext`, and other problematic macros, from our lemma. This is similar to what we did in reading the line-list file using `\dummy@ref` and various redefinitions—and that is because nested `\edtexts` macros create nested `\@ref` entries in the line-list file.

```
1328 \newcommand{\dummy@edtext}[2]{#1}
1329 %
```

`\dummy@edtext@showlemma` Some time, we want to obtain only the first argument of `\edtext`, while also wrapping it in `\showlemma`. For example, when printing a `\eledsection`.

```
1330 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1331 %
```

We are going to need another macro that takes one argument and ignores it entirely. This is supplied by the  $\TeX$  `\@gobble{<arg>}`.

`\no@expands` We need to turn off macro expansion for certain sorts of macros we are likely to see within the lemma and within the notes.

`\morenoexpands`

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.<sup>24</sup> This is done because we want to pass into our notes the generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

A second sort to turn off includes a few of the accent macros. Most are not a problem: an accent that is expanded to an `\accent` command may be harder to read but it works just the same. The ones that cause problems are: those that use alignments— $\TeX$  seems to

<sup>24</sup>Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.

get confused about the difference between alignment parameters and macro parameters; those that use temporary control sequences; and those that look carefully at what the current font is.

(The `\copyright` macro defined in PLAIN  $\TeX$  has this sort of problem as well, but is not used enough to bother with. That macro, and any other that causes trouble, will get by all right if you put a `\protect` in front of it in your file.)

We also need to eliminate all `reledmac` macros like `\edlabel` and `\setline` that write things to auxiliary files: that writing should be done only once. And we make `\edtext` itself, if it appears within its own argument, do nothing but copy its first argument.

Finally, we execute `\morenoexpands`. The version of `\morenoexpands` defined here does nothing; but you may define a version of your own when you need to add more expansion suppressions as needed with your macros. That makes it possible to make such additions without needing to copy or modify the standard `reledmac` code. If you define your own `\morenoexpands`, you must be very careful about spaces: if the macro adds any spaces to the text when it runs, extra space will appear in the main text when `\edtext` is used.

(A related problem, not addressed by these two macros, is that of characters whose category code is changed by any the macros used in the arguments to `\edtext`. Since the category codes are set when the arguments are scanned, macros that depend on changing them will not work. We have most often encountered this with characters that are made ‘active’ within text in some, but not all, of the languages used within the document. One way around the problem, if it takes this form, is to ensure that those characters are *always* active; within languages that make no special use of them, their associated control sequences should simply return the proper character. A simpler solution is to avoid active character, using Lua $\TeX$  or Xe $\TeX$ .)

```

1332 \newcommand*{\no@expands}{%
1333   \let\select@lemmafnt=0%
1334   \let\startsub=\relax \let\endsub=\relax
1335   \let\startlock=\relax \let\endlock=\relax
1336   \let\edlabel=\@gobble
1337   \let\setline=\@gobble \let\advanceline=\@gobble
1338   \let\sameword\sameword@inedtext%
1339   \let\edtext=\dummy@edtext
1340   \l@dtabnoexpands
1341   \morenoexpands}
1342 \let\morenoexpands=\relax
1343
1344 %

```

`\@tag` Now, we define an empty `\@tag` command. It will be redefine by `\edtext`: its value is the first argument. It will be used by the `\Xfootnote` commands.

```

1345 \newcommand{\@tag}{}
1346 %

```

`\@edtext@level` This counter is increased by 1 at each level of `\edtext`. That is useful for some commands which can have a different behavior if called inside or outside of the `{\langle lemma\rangle}` argument.

```
1347 \newcount\@edtext@level%
1348 \@edtext@level=0%
1349 %
```

`\theedtext` The `edtext` counter is increased at each `\edtext` command. It is used to add to insert hyperlinks between a notes and the lemma.

```
1350 \newcounter{edtext}
1351 \renewcommand{\theedtext}{edtxt@arabic{edtext}}%
1352 %
```

`\edtext` When executed, `\edtext` first ensures that we are in horizontal mode.

```
1353 \newcommand{\edtext}[2]{\leavevmode%
1354 %
```

Then, check if we are in a numbered paragraph (`\pstart...pend`).

```
1355 \ifnumberedpar%
1356 %
```

We increase the `\@edtext@level` TeX counter to know in which level of `\edtext` we are.

```
1357 \global\advance\@edtext@level by 1%
1358 %
```

We also increase the `edtext` L<sup>A</sup>T<sub>E</sub>X counter to insert `hypertarget` if the `hyperref` package is loaded.

```
1359 \stepcounter{edtext}%
1360 %
```

By default, we do not use `\lemma`

```
1361 \global\@lemmacommand@false%
1362 %
```

```
1363 \begingroup%
1364 %
```

We get the next series of `samewords` data in the list of `samewords` data for the current `edtext` level. We push them inside `\sw@inthisedtext`.

```
1365 \ifledRcol%
1366 \ifcvoid{sw@list@edtextR@the\@edtext@level}%
1367 {\global\let\sw@inthisedtext\empty}%
1368 {\expandafter\gl@p\csname sw@list@edtextR@the\@edtext@level\
endcsname\to\sw@inthisedtext}%
```



```

1369     \else%
1370         \ifcsvoid{sw@list@edtext@the\edtext@level}%
1371             {\global\let\sw@inthisedtext\empty}%
1372             {\expandafter\gl@p\csname sw@list@edtext@the\edtext@level\
\endcsname\to\sw@inthisedtext}%
1373         \fi%
1374 %

```

`\@tag` Our normal lemma is just argument #1; but that argument could have further invocations of `\edtext` within it. We get a copy of the lemma without any `\edtext` macros within it by temporarily redefining `\edtext` to just copy its first argument and ignore the other, and then expand #1 into `\@tag`, our lemma.

This is done within a group that starts here, in order to get the original `\edtext` restored; within this group we have also turned off the expansion of those control sequences commonly found within text that can cause trouble for us.

```

1375     \global\renewcommand{\@tag}{%
1376         \no@expands #1%
1377     }%
1378 %

```

`\l@d@nums` Prepare more data for the benefit of note-generating macros: the line references and font specifier for this lemma go to `\l@d@nums`.

```

1379     \set@line%
1380 %

```

`\insert@count` will be altered by the note-generating macros: it counts the number of deferred footnotes or other insertions generated by this instance of `\edtext`. If we are in a right column (`reledpar`), we use `\insert@countR` instead of `\insert@count`.

```

1381     \ifledRcol \global\insert@countR \z@%
1382     \else      \global\insert@count \z@ \fi%
1383 %

```

Now process the note-generating macros in argument #2 (i.e., `\Afootnote`, `\lemma`, etc.). `\ignorespaces` is here to skip over any spaces that might appear at the start of #2; otherwise they wind up in the main text. Footnote and other macros that are used within #2 should all end with `\ignorespaces` as well, to skip any spaces between macros when several are used in series.

```

1384     \ignorespaces #2\relax%
1385 %

```

With `polyglossia`, you must track whether the language reads left to right (English) or right to left (Arabic).

```

1386     \@ifundefined{xpg@main@language}{%if not polyglossia
1387         \flag@start}%
1388     {\if@RTL\flag@end\else\flag@start\fi%
1389     }%
1390 %

```

We write in the numbered file whether the current `\edtext` has a `\lemma` in the the second argument.

```

1391     \if@lemmacommand@%
1392     \ifledRcol%
1393     \write\linenum@outR{\string\@lemma}%
1394     \else%
1395     \write\linenum@out{\string\@lemma}%
1396     \fi%
1397     \fi%
1398 %

```

Finally, we are ready to admit the first argument into the current paragraph.

It is important that we generate and output all the notes for this chunk of text *before* putting the text into the paragraph: notes that are referenced by line number should generally be tied to the start of the passage they gloss, not the end. That should all be done within the expansion of #2 above, or in `\aftergroup` commands within that expansion.

```

1399     \endgroup%
1400     \ifdef{\hypertarget}%
1401     {%
1402     \csedef{thisedtext@the\@edtext@level}{\theedtext}%
1403     \Hy@raisedlink@left{\hypertarget{\csuse{thisedtext@the\
@edtext@level}:start}{}}%
1404     \showlemma{#1}%
1405     \Hy@raisedlink{\hypertarget{\csuse{thisedtext@the\@edtext@level}:
end}{}}%
1406     }%
1407     {%
1408     \showlemma{#1}%
1409     }%
1410 %

```

Finally, we add any insertions that are associated with the *end* of the lemma. Footnotes that are identified by symbols rather than by where the lemma begins in the main text need to be done here, and not above.

```

1411     \ifx\end@lemmas\empty \else%
1412     \glp\end@lemmas\to\x@lemma%
1413     \x@lemma%
1414     \global\let\x@lemma=\relax%
1415     \fi%
1416     \@ifundefined{xpg@main@language}{%if not polyglossia
1417     \flag@end}%
1418     {\if@RTL\flag@start\else\flag@end\fi% With polyglossia, you must
track whether the language reads left to right (English) or right to left
(Arabic).
1419     }%
1420 %

```

We switch some flags to false.

- The one that checks having footnotes inside a `\edtext`.
- The one that says we are inside a `\edtext`. In fact, it is not a flag, but a counter which is increased to 1 in each level of `\edtext`.
- The one that says we are inside a `\@lemma`.

```

1421 \global\@noneed@Footnotefalse%
1422 \global\advance\@edtext@level by -1%
1423 \global\@lemmacommand@false%
1424 %

```

If we are outside of a numbered paragraph, we send error message and print the first argument.

```

1425 \else%
1426 \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
led@err@edtextoutsidestart%
1427 \fi%
1428 }%
1429
1430 \newcommand*{\flag@end}{%
1431 \ifledRcol%
1432 \write\linenum@outR{]}%
1433 \else%
1434 \write\linenum@out{]}%
1435 \fi}%
1436
1437 %

```

`\ifnumberline` The `\ifnumberline` option can be set to `FALSE` to disable line numbering.

```

1438 \newif\ifnumberline
1439 \numberlinetrue
1440 %

```

`\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`.

One instance of `\edtext` may generate several notes, or it may generate none – it is legitimate for argument #2 to `\edtext` to be empty. But `\flag@start` and `\flag@end` induce the generation of a single entry in `\line@list` during the next run, and it is vital to also remove one and only one `\line@list` entry here.

If no more lines are listed in `\line@list`, something is wrong – probably just some change in the input. We set all the numbers to zeros, following an old publishing convention for numerical references that have not yet been resolved.

```

1441 \newcommand*{\set@line}{%
1442 \ifledRcol
1443 \ifx\line@listR\empty
1444 \global\noteschanged@true

```

```

1445 \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1446 \else
1447 \glp\line@listR\to\@tempb
1448 \xdef\l@d@nums{\@tempb|\edfont@info}%
1449 \global\let\@tempb=\undefined
1450 \fi
1451 \else
1452 \ifx\line@list\empty
1453 \global\noteschanged@true
1454 \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1455 \else
1456 \glp\line@list\to\@tempb
1457 \xdef\l@d@nums{\@tempb|\edfont@info}%
1458 \global\let\@tempb=\undefined
1459 \fi
1460 \fi}
1461
1462 %

```

`\edfont@info` The macro `\edfont@info` returns coded information about the current font.

```

1463 \newcommand*\edfont@info{\f@encoding/\f@family/\f@series/\f@shape}
1464
1465 %

```

## VI.2 Substitute lemma

`\lemma` The `\lemma{<text>}` macro allows you to change the lemma that is passed on to the notes. Read about `\@tag` in normal `\edtext` macro for more details about `\sw@list@inedtext` and `\no@expands` (VI.1 p. 113).

```

1466 \newcommand*\lemma}[1]{%
1467 \global\@lemmacommand@true%
1468 \global\renewcommand{\@tag}{%
1469 \no@expands #1%
1470 }%
1471 \ignorespaces%
1472 }%
1473 %

```

`\@lemma` The `\@lemma` is written in the numbered file to set which `\edtext` has an `\lemma` as second argument.

```

1474 \newcommand{\@lemma}{%
1475 \booltrue{lemmacommand@the\edtext@level}%
1476 }%
1477 %

```

`\if@lemmacommand@` This boolean is set to TRUE inside a `\edtext` (or `\critext`) when a `\lemma` command is called. That is useful for some commands which can have a different behavior if the lemma in the note is different from the lemma in the main text.

```
1478 \newif\if@lemmacommand@%
1479 %
```

### VI.3 Substitute line numbers

`\linenum` The `\linenum` macro can change any or all of the page and line numbers that are passed on to the notes.

As argument `\linenum` takes a set of seven parameters separated by vertical bars, in the format used internally for `\l@d@nums` (see V.9 p. 87): the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma. However, you can omit any parameters you do not want to change, and you can omit a string of vertical bars at the end of the argument. Hence `\linenum{18|4|0|18|7|1|0}` is an invocation that changes all the parameters, but `\linenum{|3}` only changes the starting line number, and leaves the rest unaltered.

We use `\\` as an internal separator for the macro parameters.

```
1480 \newcommand*{\linenum}[1]{%
1481 \xdef\@tempa{#1|||||}\noexpand\\l@d@nums}%
1482 \global\let\l@d@nums=\empty
1483 \expandafter\line@set\@tempa|\\ignorespaces}
1484 %
```

`\line@set` `\linenum` calls `\line@set` to do the actual work; it looks at the first number in the argument to `\linenum`, sets the corresponding value in `\l@d@nums`, and then calls itself to process the next number in the `\linenum` argument, if there are more numbers in `\l@d@nums` to process.

```
1485 \def\line@set#1|#2|#3|#4|{%
1486 \gdef\@tempb{#1}%
1487 \ifx\@tempb\empty
1488 \l@d@add{#3}%
1489 \else
1490 \l@d@add{#1}%
1491 \fi
1492 \gdef\@tempb{#4}%
1493 \ifx\@tempb\empty\else
1494 \l@d@add{|}\line@set#2|#4|}%
1495 \fi}
1496 %
```

`\l@d@add` `\line@set` uses `\l@d@add` to tack numbers or vertical bars onto the right hand end of `\l@d@nums`.

```
1497 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1498
1499 %
```

## VI.4 Lemma disambiguation

The mechanism which counts the occurrence of a same word in a same line is quite complex, because, when  $\text{\LaTeX}$  reads a command between a  $\text{\pstart}$  and a  $\text{\pend}$ , it does not know yet which are the line numbers.

The general mechanism is the following:

- **At the first run**, each  $\text{\sameword}$  command increments an  $\text{etoolbox}$  counter the name of which contains the argument of the  $\text{\sameword}$  commands.
- Then this counter, associated with the argument of  $\text{\sameword}$  is stored, with the  $\text{\@sw}$  command, in the auxiliary file of the current  $\text{eledmac}$  section (the  $\text{.1}$ ,  $\text{.2}$ ... file).
- **When this auxiliary file is read at the second run**, different operations are achieved:

1. Get the rank of each  $\text{\sameword}$  in a line (relative rank) from the rank of each  $\text{\sameword}$  in all the numbered section (absolute rank):

- For each paired  $\text{\sameword}$  argument and absolute line number, a counter is defined. Its value corresponds to the number of times  $\text{\sameword}\{\langle argument \rangle\}+$  is called from the beginning of the lineation to the end of the current line. We also store the same data for the preceding absolute line number, if it does not have  $\text{\sameword}\{\langle argument \rangle\}$ .
- For each  $\text{\sameword}$  having the same argument, we subtract from its absolute rank the number stored for the paired  $\text{\sameword}$  argument and previous absolute line number. Consequently, we obtain the relative rank.
- See the following example which explain how for same  $\text{\sameword}$  absolute ranks are transformed to relative rank.

```
At line 1:
absolute rank 1 becomes relative rank 1-0 = 1
1 is stored for this \sameword and the line 1
At line 2:
absolute rank 2 becomes relative rank 2-1 = 1
absolute rank 3 becomes relative rank 3-2 = 2
3 is stored for this \sameword and the line 2
At line 3:
no \sameword for this line.
3 is stored for this \sameword and the line 3
At line 4:
absolute rank 4 becomes relative rank 4-3 = 1
3 is stored for this \sameword and the line 4
```

2. Create lists of lists of  $\text{\sameword}$  by depth of  $\text{\edtext}$ . That is: create a list for  $\text{\edtext}$  of level 1, a list for  $\text{\edtext}$  of level 2, a list for  $\text{\edtext}$  of level 3 etc. For each  $\text{\edtext}$  in these list, we store all the relative rank of  $\text{\sameword}$  which are called as lemma information, that is 1) or called

in the first argument of `\sameword 2`) or called in the `\lemma` macro of the second argument of `\sameword AND` marked by the optional argument of `\sameword` in first argument of `\edtext`.

For example, suppose a line with nested `\edtexts` which contains some word marked by `\sameword` and having the following relative rank:

bar<sup>1</sup> foo<sup>1</sup> foo<sup>2</sup> bar<sup>2</sup> foo<sup>3</sup> (A)(B) foo<sup>4</sup> bar<sup>3</sup> (C) foo<sup>5</sup> (D) bar<sup>4</sup> (E)

In this example, all lemma information for `\edtext` is framed. The text in parenthesis is the content of critical notes associated to the preceding frame. As you can see, we have two level of `\edtext`.

The list for `\edtexts` of level 1 is  $\{\{1, 2, 2, 3, 4, 3\}, \{5, 4\}\}$ .

The list for `\edtexts` of level 2 is  $\{\{1, 2, 2, 3\}, \{5\}\}$ .

As you can see, the mandatory argument of `\sameword` does not matter: we store the rank informations for every word potentially ambiguous.

- At the second run, when a critical notes is called, we associate it to the next item of the list associated to is `\edtext` level. So, in the previous example:
  - Critical notes (A) and (B) are associated with  $\{1, 2, 2, 3\}$ .
  - Critical note (C) is associated with  $\{1, 2, 2, 3, 4, 3\}$ .
  - Critical note (D) is associated with  $\{5\}$ .
  - Critical note (E) is associated with  $\{5, 4\}$ .
- At the second run, when a critical note is printed:
  - The `\sameword` command is let `\sameword@inedtext`.
  - At each call of this `\sameword@inedtext`, we step to the next element of the list associated to the note. Let it be  $r$ .
  - For the word marked by `\sameword`, we calculate how many time it is called in its line. To do it:
    - \* We get the absolute line number of the current `\sameword`. This absolute line number was stored with list of relative rank for the current `\edtext`. That means, in the previous example, that, if the absolute line number of `\edtext` was 1, that critical notes (A) and (B) were not associated with  $\{1, 2, 2, 3\}$  but with  $\{(1, 1), (2, 1), (2, 1), (3, 1)\}$ . Such method to know the absolute line number associated to a `\sameword` is required because a `\edtext` can be overlap many lines, but `\sameword` can't get it.
    - \* We get the value associated, when reading the auxiliary file, to the pair compose by the current marked word and the current absolute line number. Let this value be  $n$ .
  - If  $n > 1$ , that mean the current word appears more than once time in its line. In this case, we call `\showwordrank` with the word as first argument and  $r$  as second argument. If the word is called only once, we just print it.

After theory, implementation.

`\get@sw@txt` As the argument of `\sameword` can contain active character if we use `inputenc` with `utf8` option instead of native UTF-8 engine, we store its detokenized content in a macro in order to allow dynamic name of macro with `\csname`.<sup>25</sup>

Because there is a bug with `\detokenize` and  $X_{\text{TeX}}$  when using non BMP characters<sup>26</sup>, we detokenize only for not  $X_{\text{TeX}}$  engines. In any case, in  $X_{\text{TeX}}$ , a `\csname` construction can contain UTF-8 characters without a problem, as UTF-8 characters are not managed with category code, but instead read directly as UTF-8 characters.

```

1500 \newcommand{\get@sw@txt}[1]{%
1501   \ifxetex%
1502     \xdef\sw@txt{#1}%
1503   \else%
1504     \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
1505   \fi%
1506 }%
1507 %

```

`\sameword` The high level macro `\sameword`, used by the editor.

```

1508 \newcommandx{\sameword}[2][1,usedefault]{%
1509   \leavevmode%
1510   \get@sw@txt{#2}%
1511 %

```

Now, the real code. First, increment the counter corresponding to the argument.

```

1512   \unless\ifledRcol%
1513     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1514 %

```

Then, write its value to the numbered file.

```

1515     \protected@write\linenum@out{\string\@sw{\sw@txt}{\csuse{sw@\sw@txt}
1516 }}{#1}}%

```

Do the same thing if we are in the right columns.

```

1517   \else%
1518     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1519     \protected@write\linenum@outR{\string\@sw{\sw@txt}{\csuse{sw@\sw@txt}
1520 }}{#1}}%
1521   \fi%

```

And print the word.

```

1522   #2%
1523 }%
1524 %

```

<sup>25</sup>See <http://tex.stackexchange.com/q/244538/7712>.

<sup>26</sup><http://sourceforge.net/p/xetex/bugs/108/>



A flag set to true if a `\@sw` relative rank must be added to the list of ranks for a specific `\edtext`.

```
\if@addsw25 \newif\if@addsw%
1526 %
```

`\@sw` The command printed in the auxiliary files.

```
1527 \newcommand{\@sw}[3]{%
1528   \get@sw@txt{#1}%
1529   \unless\ifledRcol%
1530 %
```

First, define a counter which store the second argument as value for a each paired absolute line number/first argument

```
1531   \csxdef{sw@\sw@txt @\the\absline@num @\the\section@num}{#2}%
1532 %
```

If such argument was not defined for the preceding line, define it.

```
1533   \numdef{\prev@line}{\the\absline@num-1}%
1534   \ifcsundef{sw@\sw@txt @\prev@line @\the\section@num}{%
1535     \csnumgdef{sw@\sw@txt @\prev@line @\the\section@num}{#2-1}%
1536   }{}%
1537 %
```

Then, calculate the position of the word in the line.

```
1538   \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1539 %
```

And do the same thing for the right side.

```
1540   \else%
1541     \csxdef{sw@\sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1542     \numdef{\prev@line}{\the\absline@numR-1}%
1543     \ifcsundef{sw@\sw@txt @\prev@line @\the\section@numR @R}{%
1544       \csnumgdef{sw@\sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1545     }{}%
1546     \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1547   \fi%
1548 %
```

And now, add it to the list of `\@sw` for the current `\edtext`, in all depth.

```
1549   \@tempcnta=\@edtext@level
1550   \@whilenum{\@tempcnta>0}\do{%
1551     \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1552     {%
1553       \@addswfalse%
1554       \notbool{lemmacommand@\the\@tempcnta}%
1555     }{\@addswtrue}%
```

```

1556     {\IfStrEq{#3}{inlemma}%
1557      {\@addswtrue}%
1558      {%
1559       \def\do##1{%
1560         \ifnumequal{##1}{\the\@tempcnta}%
1561         {\@addswtrue\listbreak}%
1562         }%
1563       }%
1564       \docsvlist{#3}%
1565     }%
1566   }%
1567   \if@addsw%
1568     \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1569     \ifledRcol%
1570       \xright@appenditem{\the@sw}{\the\absline@numR}}\to\@tmp%
1571     \else%
1572       \xright@appenditem{\the@sw}{\the\absline@num}}\to\@tmp%
1573     \fi%
1574     \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1575     \fi%
1576   }%
1577   }%
1578   \advance\@tempcnta by -1%
1579 }%
1580 }%
1581 %

```

`\sameword@inedtext` The command called when `\sameword` is called in a `\edtext`.

```

1582 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1583   \get@sw@txt{#2}%
1584   \unless\ifledRcol%
1585 %

```

Just a precaution.

```

1586   \ifx\sw@list@inedtext\empty%
1587     \def\the@sw{999}%
1588     \def\this@absline{-99}%
1589   \else%
1590 %

```

But in many cases, at this step, we should have some content in the list `\sw@list@inedtext`, which contains the reference for `\edtext`.

```

1591   \gl@p\sw@list@inedtext\to\@tmp%
1592   \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1593   \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1594   \fi%
1595 %

```

First, calculate the number of occurrences of the word in the current line

```

1596 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
1597 \numdef{\prev@line}{\this@absline-1}%
1598 \numdef{\sw@atthisline}{\cuse{sw@\sw@txt @\this@absline @\the\
section@num}-\cuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1599 }%
1600 {\numdef{\sw@atthisline}{0}}%
1601 %

```

Finally, print the rank, but only if there is more than one occurrence of the word in the current line.

```

1602 \ifnumgreater{\sw@atthisline}{1}%
1603 {\showwordrank{#2}{\the@sw}}%
1604 {#2}%
1605 %

```

And the same for right side.

```

1606 \else%
1607 \ifx\sw@list@inedtext\empty%
1608 \def\the@sw{999}%
1609 \def\this@absline{-99}%
1610 \else%
1611 \gl@p\sw@list@inedtext\to\@tmp%
1612 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1613 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1614 \fi%
1615 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
1616 \numdef{\prev@line}{\this@absline-1}%
1617 \numdef{\sw@atthisline}{\cuse{sw@\sw@txt @\this@absline @\the\
section@numR @R}-\cuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1618 }%
1619 {\numdef{\sw@atthisline}{0}}%
1620 \ifnumgreater{\sw@atthisline}{1}%
1621 {\showwordrank{#2}{\the@sw}}%
1622 {#2}%
1623 \fi%
1624 }%
1625 %

```

```

\showwordrank#26 % Finally, the way the rank will be printed.

```

```

1627 \newcommand{\showwordrank}[2]{%
1628 #1\textsuperscript{#2}%
1629 }%
1630 %

```

## VII 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.

### VII.1 Boxes, counters, `\pstart` and `\pend`

`\raw@text` Here are numbers and flags that are used internally in the course of the paragraph decomposition.  
`\ifnumberedpar@`  
`\numberedpar@true` When we first form the paragraph, it goes into a box register, `\raw@text`, instead  
`\numberedpar@false` of onto the current vertical list. The `\ifnumberedpar@` flag will be true while a para-  
`\num@lines` graph is being processed in that way. `\num@lines` will store the number of lines in the  
`\one@line` paragraph when it is complete. When we chop it up into lines, each line in turn goes  
`\par@line` into the `\one@line` register, and `\par@line` will be the number of that line within the  
paragraph.

```
1631 \newbox\raw@text
1632 \newif\ifnumberedpar@
1633 \newcount\num@lines
1634 \newbox\one@line
1635 \newcount\par@line
1636 %
```

`\pstart` `\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant  
`\AtEveryPstart` variables, and then arranges for the subsequent text to go into the `\raw@text` box.  
`\numberpstarttrue` `\pstart` needs to appear at the start of every paragraph that is to be numbered; the  
`\numberpstartfalse` `\autopar` command below may be used to insert these commands automatically.

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

```
1637
1638 \newcommand{\AtEveryPstart}[1]{%
1639   \ifstrempy{#1}%
1640     {\xdef\at@every@pstart{}}%
1641     {\gdef\at@every@pstart{\noindent#1}}%
1642 }%
1643 \xdef\at@every@pstart{}%
1644
1645 \newcounter{pstart}
1646 \renewcommand{\thepstart}{\bfseries\@arabic@c@pstart}. }
1647 \newif\ifnumberpstart
1648 \numberpstartfalse
1649 \newif\iflabelpstart
1650 \labelpstartfalse
1651 \newcommandx*{\pstart}[1][1]{%
1652   \normal@pars%
1653   \ifstrempy{#1}{\at@every@pstart}{\noindent#1}%
1654   \ifluatex%
```

```

1655 \edef\l@luatexttextdir@L{\the\textdir}%
1656 \fi%
1657 \@nobraektrue%
1658 \ifnumbering \else%
1659 \led@err@PstartNotNumbered%
1660 \beginnumbering%
1661 \fi%
1662 \ifnumberedpar@%
1663 \led@err@PstartInPstart%
1664 \pend%
1665 \fi%
1666 \list@clear{\inserts@list}%
1667 \global\let\next@insert=\empty%
1668 \begingroup%
1669 \global\advance \l@dnumpstartsL\@ne
1670 \global\setbox\raw@text=\vbox\bgroup%
1671 \ifautopar\else%
1672 \ifnumberpstart%
1673 \ifinstanza\else%
1674 \ifsidepstartnum\else%
1675 \thepstart%
1676 \fi%
1677 \fi%
1678 \fi%
1679 \fi%
1680 \numberedpar@true%
1681 \iflabelpstart\protected@edef\@currentlabel%
1682 {\p@pstart\thepstart}
1683 \fi%
1684 \l@dzeropenalties%
1685 \ignorespaces%because not automatically ignored if an optional argument
is used (classical TeX behavior for space after commands)
1686 }
1687 %

```

`\pend` `\pend` must be used to end a numbered paragraph.

```

1688 \newcommand*{\pend}[1][1]{\ifnumbering \else%
1689 \led@err@PendNotNumbered%
1690 \fi%
1691 \global\l@dskipversenumberfalse%
1692 \ifnumberedpar@ \else%
1693 \led@err@PendNoPstart%
1694 \fi%
1695 %

```

We set all the usual interline penalties to zero and then 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. These penalties revert to the values that you set when the group for the `\vbox` ends. Then we call `\do@line` to slice a line off the top

of the paragraph, add a line number and footnotes, and restore it to the page; we keep doing this until there are not any more lines left.

```

1696 \l@dzero penalties%
1697 \endgraf\global\num@lines=\prevgraf\egroup%
1698 \global\par@line=0%
1699 %

```

We check if lineation is by `pstart`: in this case, we reset line number, but only in the second line of the `pstart`. We can't reset line number at the beginning of `\pstart`, as `\setline` is parsed at the end of previous `\pend`, and so, we must do it at the end of first line of `pstart`.

```

1700 \csnumdef{pstartline}{0}%
1701 \loop\ifvbox\raw@text%
1702 \csnumdef{pstartline}{\pstartline+1}%
1703 \do@line%
1704 \ifbypstart@%
1705 \ifnumequal{\pstartline}{1}{%
1706 \bgroup%
1707 \let\leavevmode\relax%
1708 \setline{1}%
1709 \egroup%
1710 \resetprevline@}{}%
1711 \fi%
1712 \repeat%
1713 %

```

Deal with any leftover notes, and then end the group that was begun in the `\pstart`.

```

1714 \flush@notes%
1715 \endgroup%
1716 \ignorespaces%
1717 %

```

Increase `pstart` counter.

```

1718 \ifnumberpstart%
1719 \pstartnumtrue%
1720 \fi%
1721 \addtocounter{pstart}{1}%
1722 %

```

Print the optional argument of `\pend` or the content printed after every `\pend`

```

1723 \normal@pars%
1724 \ifstrempy{#1}{\at@every@pend}{\noindent#1}%
1725 %

```

Restore standard `nobreak` setting and `autopar` setting. Normally, `\if@nobreak` is equal to `true` only immediately after a sectioning command (read `latex.ltx` file). As a `\pstart... \pend` structure can't contain any sectioning command, we set `\if@nobreak` to `false`.

```

1726 \nobeakfalse%
1727 \ifautopar%
1728   \autopar%
1729 \fi%
1730 }
1731
1732 %

```

Here, two macros to insert content after every `\pend`, between numbered line. `\AtEveryPend` is the user macro, `\at@every@pend` is macro set by it.

```

\AtEveryPend33
\at@every@pend34 \newcommand{\AtEveryPend}[1]{%
1735   \ifstrempy{#1}%
1736   {\xdef\at@every@pend{}}%
1737   {\gdef\at@every@pend{\noindent#1}}%
1738 }%
1739 \xdef\at@every@pend{}%
1740
1741 %

```

`\l@dzeropenalties` A macro to zero penalties for `\pend` or `\pstart`.

```

1742 \newcommand*\l@dzeropenalties{%
1743   \brokenpenalty \z@ \clubpenalty \z@
1744   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1745   \postdisplaypenalty \z@ \widowpenalty \z@}
1746
1747 %

```

`\autopar` In most cases it is only an annoyance to have to label the paragraphs to be numbered with `\pstart` and `\pend`. `\autopar` will do that automatically, allowing you to start a paragraph with its first word and no other preliminaries, and to end it with a blank line or a `\par` command. The command should be issued within a group, after `\beginnumbering` has been used to start the numbering; all paragraphs within the group will be affected.

A few situations can cause problems. One is a paragraph that begins with a begin-group character or command: `\pstart` will not get invoked until after such a group beginning is processed; as a result the character that ends the group will be mistaken for the end of the `\vbox` that `\pstart` creates, and the rest of the paragraph will not be numbered. Such paragraphs need to be started explicitly using `\indent`, `\noindent`, or `\leavevmode` — or `\pstart`, since you can still include your own `\pstart` and `\pend` commands even with `\autopar` on.

Prematurely ending the group within which `\autopar` is in effect will cause a similar problem. You must either leave a blank line or use `\par` to end the last paragraph before you end the group.

The functioning of this macro is more tricky than the usual `\everypar`: we do not want anything to go onto the vertical list at all, so we have to end the paragraph, erase any evidence that it ever existed, and start it again using `\pstart`. We remove the paragraph-indentation box using `\lastbox` and save the width, and then skip backwards over the `\parskip` that has been added for this paragraph. Then we start again with `\pstart`, restoring the indentation that we saved, and locally change `\par` so that it will do our `\pend` for us.

```

1748 \newif\ifautopar
1749 \autoparfalse
1750 \newcommand*{\autopar}{
1751   \ifledRcol
1752     \ifnumberingR \else
1753       \led@err@AutoparNotNumbered
1754       \beginnumberingR
1755       \fi
1756     \else
1757       \ifnumbering \else
1758         \led@err@AutoparNotNumbered
1759         \beginnumbering
1760         \fi
1761       \fi
1762     \autopartrue
1763     \everypar{\setbox0=\lastbox
1764               \endgraf \vskip-\parskip
1765               \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\fi\fi
1766     \let\par=\pend}%
1767   \ignorespaces}
1768 %

```

`\normal@pars` We also define a macro which we can rely on to turn off the `\autopar` definitions at various important places, if they are in force. We will want to do this within a footnotes, for example.

```

1769 \newcommand*{\normal@pars}{\ifautopar\everypar{}\fi\let\par\endgraf}
1770
1771 %

```

`\ifautopar@pause` We define a boolean test switched to true at the beginning of the `\pausenumbering` command if the autopar is enabled. This boolean will be tested at the beginning of `\resumenumbering` to continue the autopar if needed.

```

1772 \newif\ifautopar@pause
1773 %

```



## VII.2 Processing one line

### VII.2.1 General process

`\do@line` The `\do@line` macro is called by `\pend` to do all the processing for a single line of text.  
`\l@dunhbox@line`

```

1774 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1775 \newcommand*{\do@line}{%
1776   {\vbadness=10000
1777    \splittopskip=\z@
1778    \do@linehook
1779   \l@demptyd@ta
1780    \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1781   \unvbox\one@line \global\setbox\one@line=\lastbox
1782   \getline@num
1783   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{-}
1784   \ifnum\@lock>\@ne
1785     \inserthangingsymboltrue
1786   \else
1787     \inserthangingsymbolfalse
1788   \fi
1789   \check@pb@in@verse
1790   \ifl@dhiddenumber%
1791     \global\l@dhiddenumberfalse%
1792   \fix@locks%
1793   \else%
1794     \affixline@num%
1795   \fi%
1796 %

```

Depending whether a sectioning command is called at this pstart or not we print sectioning command or normal line,

```

1797 \xifinlist{\the\l@dnumpstartsL}{\eled@sections@}{%
1798   {\print@eledsection}%
1799   {\print@line}%
1800   \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{-}
1801 }%
1802 %

```

### VII.2.2 Process for “normal” line

`\print@line` `\print@line` is for normal line, i. e. line without sectioning command.

```

1803 \def\print@line{
1804 %

```

Insert the pstart number in side, if we are in the first line of a pstart.

```

1805   \affixpstart@num%
1806 %

```

The line will be boxed, to have the good width.

```
1807 \hb@xt@ \linewidth{%
1808 %
```

User hook.

```
1809 \do@insidelinehook%
1810 %
```

Left line number

```
1811 \l@dld@ta%
1812 %
```

Restore marginal and footnotes.

```
1813 \add@inserts\affixside@note%
1814 %
```

Print left notes.

```
1815 \l@dlsn@te
1816 %
```

Boxes the line, writes information about new line in the numbered file.

```
1817 {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1818 %
```

If we use Lua $\TeX$  then restore the direction.

```
1819 \ifluatex%
1820 \textdir\l@luatextextdir@L%
1821 \fi%
1822 %
```

Insert, if needed, the hanging symbol.

```
1823 \inserthangingsymbol %Space kept for backward compatibility
1824 %
```

And so, print the line.

```
1825 \l@dunhbox@line{\one@line}}%
1826 %
```

Right line number

```
1827 \ledrlfill\l@drd@ta%
1828 %
```

Print right notes.

```
1829 \l@drsn@te
1830 }}%
1831 %
```

And reinsert penalties (for page breaking)...

```
1832 \add@penalties%
1833 }
1834 %
```

### VII.2.3 Process for line containing \eledsection command

`\print@eledsection` `\print@eledsection` to print sectioning command with line number. It sets the correct spacing, depending whether a sectioning command was called at previous `\pstart`, calls the sectioning command, prints the normal line outside of the paper, to be able to have critical footnotes. Because of how this prints, a vertical spacing correction is added.

```

1835 \def\print@eledsection{%
1836   \add@inserts\affixside@note%
1837   \numdef{\temp@}{\l@dnumpstartsL-1}%
1838   \xifinlist{\temp@}{\eled@sections@}{\@nobreaktrue}{\@nobreakfalse}%
1839   \@eled@sectioningtrue%
1840   \csuse{eled@sectioning@the\l@dnumpstartsL}%
1841   \@eled@sectioningfalse%
1842   \global\csundef{eled@sectioning@the\l@dnumpstartsL}%
1843   \if@RTL%
1844     \hspace{-3\paperwidth}%
1845     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1846   \else%
1847     \hspace{3\paperwidth}%
1848     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1849   \fi%
1850   \vskip-\baselineskip%
1851 }
1852 %

```

### VII.2.4 Hooks

`\do@linehook` `\do@insidelinehook` Two hooks into `\do@line`. The first is called at the beginning of `\do@line`, the second is called in the line box. The second can, for example, have a `\markboth` command inside, the first can not.

```

1853 \newcommand*{\do@linehook}{}
1854 \newcommand*{\do@insidelinehook}{}
1855 %

```

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

```

1856 \newcommand*{\dolinehook}[1]{\gdef\do@linehook{#1}}%
1857 \newcommand*{\doinsidelinehook}[1]{\gdef\do@insidelinehook{#1}}%
1858
1859 %

```

### VII.2.5 Sidenotes and marginal line number initialization

`\l@emptyd@ta` `\l@dld@ta` `\l@drd@ta` `\l@dcsnotetext` `\l@dcsnotetext@l` `\l@dcsnotetext@r` Nulls the `\. . .d@ta`, which may later hold line numbers. Similarly for `\l@dcsnotetext`, `\l@dcsnotetext@l`, `\l@dcsnotetext@r` for the texts of the sidenotes, left and right notes.

```

1860 \newcommand*\l@emptyd@ta}{%
1861   \gdef\l@dld@ta}{%
1862   \gdef\l@drd@ta}{%
1863   \gdef\l@dcsnotetext@l}{%
1864   \gdef\l@dcsnotetext@r}{%
1865   \gdef\l@dcsnotetext{}%
1866 %
1867 %

```

**\l@dlsn@te** Zero width boxes of the left and right side notes, together with their kerns.

**\l@drsn@te**

```

1868 \newcommand*\l@dlsn@te}{%
1869   \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1870 \newcommand*\l@drsn@te}{%
1871   \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1872 %
1873 %

```

**\ledllfill** These macros are called at the left (**\ledllfill**) and the right (**\ledllfill**) of each numbered line. The initial definitions correspond to the original code for **\do@line**.

**\ledrlfill**

```

1874 \newcommand*\ledllfill}{\hfil}
1875 \newcommand*\ledrlfill}{%
1876 %
1877 %

```

## VIII Line and page number computation

**\getline@num** The **\getline@num** macro determines the page and line numbers for the line we are about to send to the vertical list.

```

1878 \newcommand*\getline@num}{%
1879   \global\advance\absline@num \@ne%
1880   \do@actions
1881   \do@ballast
1882   \ifnumberline
1883     \ifsublines@
1884       \ifnum\sub@lock<\tw@
1885         \global\advance\subline@num \@ne
1886       \fi
1887     \else
1888       \ifnum\@lock<\tw@
1889         \global\advance\line@num \@ne
1890         \global\subline@num \z@
1891       \fi
1892     \fi
1893   \fi
1894 }%
1895 %

```

`\do@ballast` The real work in the macro above is done in `\do@actions`, but before we plunge into that, let's get `\do@ballast` out of the way. This macro looks to see if there is an action to be performed on the *next* line, and if it is going to be a page break action, `\do@ballast` decreases the count `\ballast@count` counter by the amount of `ballast`. This means, in practice, that when `\add@penalties` assigns penalties at this point,  $\TeX$  will be given extra encouragement to break the page here (see XI.2 p. 142).

`\ballast@count` First we set up the required counters; they are initially set to zero, and will remain so  
`\c@ballast` unless you type `\setcounter{ballast}{some figure}` in your document.

```
1896 \newcount\ballast@count
1897 \newcounter{ballast}
1898 \setcounter{ballast}{0}
1899 %
```

And here is `\do@ballast` itself. It advances `\absline@num` within the protection of a group to make its check for what happens on the next line.

```
1900 \newcommand*\do@ballast{\global\ballast@count \z@
1901 \begingroup
1902 \advance\absline@num \@ne
1903 \ifnum\next@actionline=\absline@num
1904 \ifnum\next@action>-1001\relax
1905 \global\advance\ballast@count by -\c@ballast
1906 \fi
1907 \fi
1908 \endgroup}
1909 %
```

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

It may call itself recursively, and to do this efficiently (using  $\TeX$ 's optimization for tail recursion), we define a control-sequence called `\do@actions@next` that is always the last thing that `\do@actions` does. If there could be more actions to process for this line, `\do@actions@next` is set equal to `\do@actions`; otherwise it is just `\relax`.

```
1910 \newcommand*\do@actions}{%
1911 \global\let\do@actions@next=\relax
1912 \ifnum\absline@num<\next@actionline\else
1913 %
```

First, page number changes, which will generally be the most common actions. If we are restarting lineation on each page, this is where it happens.

```
1914 \ifnum\next@action>-1001
1915 \global\page@num=\next@action
1916 \ifbypage@
1917 \global\line@num=\z@ \global\subline@num=\z@
1918 \resetprevline@
1919 \fi
1920 %
```

Next, we handle commands that change the line-number values. (We subtract 5001 rather than 5000 here because the line number is going to be incremented automatically in `\getline@num`.)

```

1921 \else
1922   \ifnum\next@action<-4999
1923     \@l@dttempcnta=-\next@action
1924     \advance\@l@dttempcnta by -5001
1925     \ifsublines@
1926       \global\subline@num=\@l@dttempcnta
1927     \else
1928       \global\line@num=\@l@dttempcnta
1929     \fi
1930 %

```

We rescale the value in `\@l@dttempcnta` so that we can use a case statement.

```

1931 \else
1932   \@l@dttempcnta=-\next@action
1933   \advance\@l@dttempcnta by -1000
1934   \do@actions@fixedcode
1935 \fi
1936 \fi
1937 %

```

Now we get information about the next action off the list, and then set `\do@actions@next` so that we will call ourselves recursively: the next action might also be for this line.

There is no warning if we find `\actionlines@list` empty, since that will always happen near the end of the section.

```

1938 \ifx\actionlines@list\empty
1939   \gdef\next@actionline{1000000}%
1940 \else
1941   \gl@p\actionlines@list\to\next@actionline
1942   \gl@p\actions@list\to\next@action
1943   \global\let\do@actions@next=\do@actions
1944 \fi
1945 \fi
1946 %

```

Make the recursive call, if necessary.

```

1947 \do@actions@next}
1948
1949 %

```

`\do@actions@fixedcode` This macro handles the fixed codes for `\do@actions`. It is one big case statement.

```

1950 \newcommand*{\do@actions@fixedcode}{%
1951   \ifcase\@l@dttempcnta
1952   \or% % 1001
1953   \global\sublines@true

```

```

1954 \or% % 1002
1955 \global\sublines@false
1956 \or% % 1003
1957 \global\@lock=\@ne
1958 \or% % 1004
1959 \ifnum\@lock=\tw@
1960 \global\@lock=\thr@@
1961 \else
1962 \global\@lock=\z@
1963 \fi
1964 \or% % 1005
1965 \global\sub@lock=\@ne
1966 \or% % 1006
1967 \ifnum\sub@lock=\tw@
1968 \global\sub@lock=\thr@@
1969 \else
1970 \global\sub@lock=\z@
1971 \fi
1972 \or% % 1007
1973 \l@dskipnumbertrue
1974 \or% % 1008
1975 \l@dskipversenumbertrue%
1976 \or% % 1009
1977 \l@dhiddenumbertrue
1978 \else
1979 \led@warn@BadAction
1980 \fi}
1981
1982
1983 %

```

## IX Line number printing

`\affixline@num` `\affixline@num` just puts a left line number into `\l@dld@ta` or a right line number into `\l@drd@ta` if required.

To determine whether we need to affix a line number to this line, we compute the following:

$$\begin{aligned}
 n &= \text{int}((\text{linenum} - \text{firstlinenum}) / \text{linenumincrement}) \\
 m &= \text{firstlinenum} + (n \times \text{linenumincrement})
 \end{aligned}$$

(where *int* truncates a real number to an integer). *m* will be equal to *linenum* only if we are to paste a number on here. However, the formula breaks down for the first line to number (and any before that), so we check that case separately: if `\line@num`  $\leq$  `\firstlinenum`, we compare the two directly instead of making these calculations.

We compute, in the scratch counter `\@l@tempcnta`, the number of the next line that should be printed with a number (*m* in the above discussion), and move the current line number into the counter `\@l@tempcntb` for comparison.

First, the case when we are within a sub-line range.

```
1984 \newcommand*\affixline@num}{%
1985 %
```

No number is attached if `\ifl@dskipnumber` is TRUE (and then it is set to its normal FALSE value). No number is attached if `\ifnumberline` is FALSE (the normal value is TRUE).

```
1986 \ifledgroupnotesL@else
1987 \ifnumberline
1988 \ifl@dskipnumber
1989 \global\l@dskipnumberfalse
1990 \else
1991 \ifsublines@
1992 \l@dttempcntb=\subline@num
1993 \ifnum\subline@num>\c@firstsublinenum
1994 \l@dttempcnta=\subline@num
1995 \advance\l@dttempcnta by-\c@firstsublinenum
1996 \divide\l@dttempcnta by\c@sublinenumincrement
1997 \multiply\l@dttempcnta by\c@sublinenumincrement
1998 \advance\l@dttempcnta by\c@firstsublinenum
1999 \else
2000 \l@dttempcnta=\c@firstsublinenum
2001 \fi
2002 %
```

That takes care of computing the values for comparison, but if line number locking is in effect we have to make a further check. If this check fails, then we disable the line-number display by setting the counters to arbitrary but unequal values.

```
2003 \ch@cksub@l@ck
2004 %
```

Now the line number case, which works the same way.

```
2005 \else
2006 \l@dttempcntb=\line@num
2007 %
```

Check on the `\linenumberlist` If it is `\empty` use the standard algorithm.

```
2008 \ifx\linenumberlist\empty
2009 \ifnum\line@num>\c@firstlinenum
2010 \l@dttempcnta=\line@num
2011 \advance\l@dttempcnta by-\c@firstlinenum
2012 \divide\l@dttempcnta by\c@linenumincrement
2013 \multiply\l@dttempcnta by\c@linenumincrement
2014 \advance\l@dttempcnta by\c@firstlinenum
2015 \else
2016 \l@dttempcnta=\c@firstlinenum
2017 \fi
2018 \else
2019 %
```



The `\linenumberlist` was not `\empty`, so here is Wayne's numbering mechanism. This takes place in  $\TeX$ 's mouth.

```

2020     \l@ldtempcnta=\line@num
2021     \edef\rem@inder{\linenumberlist,\number\line@num,}%
2022     \edef\sc@n@list{\def\noexpand\sc@n@list
2023     #####1,\number\l@ldtempcnta,#####2|\def\noexpand\rem@inder
2024     {#####2}}}%
2025     \sc@n@list\expandafter\sc@n@list\rem@inder|
2026     \ifx\rem@inder\empty%
2027     \advance\l@ldtempcnta\@ne
2028     \fi
2029 %

```

A locking check for lines, just like the version for sub-line numbers above.

```

2030     \ch@ck@l@ck
2031     \fi
2032 %

```

The following tests are true if we need to print a line number.

```

2033     \ifnum\l@ldtempcnta=\l@ldtempcntb
2034     \ifl@dskipversenumber\else
2035 %

```

If we got here, we are going to print a line number; so now we need to calculate a number that will tell us which side of the page will get the line number. We start from `\line@margin`, which asks for one side always if it is less than 2; and then if the side does depend on the page number, we simply add the page number to this side code—because the values of `\line@margin` have been devised so that this produces a number that is even for left-margin numbers and odd for right-margin numbers.

For  $\TeX$  we have to consider two column documents as well. In this case Peter Wilson thought we need to put the numbers at the outside of the column — the left of the first column and the right of the second. Do the `twocolumn` stuff before going on with the original code.

`\l@dld@ta` A left line number is stored in `\l@dld@ta` and a right one in `\l@drd@ta`.

```

2036     \if@twocolumn
2037     \if@firstcolumn
2038     \gdef\l@dld@ta{\llap{\leftlinenum}}%
2039     \else
2040     \gdef\l@drd@ta{\rlap{\rightlinenum}}%
2041     \fi
2042     \else
2043     \l@ldtempcntb=\line@margin
2044     \ifnum\l@ldtempcntb>\@ne
2045     \advance\l@ldtempcntb \page@num
2046     \fi
2047     \ifodd\l@ldtempcntb

```

```

2048         \gdef\l@drd@ta{\rlap{{\rightlinenum}}}%
2049         \else
2050         \gdef\l@dld@ta{\llap{{\leftlinenum}}}%
2051         \fi
2052     \fi
2053 \fi
2054 \fi
2055 %

```

Now fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

2056     \f@x@l@cks
2057     \fi
2058     \fi
2059     \fi
2060 }
2061 %
2062 %

```

`\ch@cksub@l@ck` These macros handle line number locking for `\affixline@num`. `\ch@cksub@l@ck` checks subline locking. If it fails, then we disable the line-number display by setting the counters to arbitrary but unequal values.

```

2063 \newcommand*{\ch@cksub@l@ck}{%
2064     \ifcase\sub@lock
2065     \or
2066     \ifnum\sublock@disp=\@ne
2067     \l@dttempcntb=\z@ \l@dttempcnta=\@ne
2068     \fi
2069     \or
2070     \ifnum\sublock@disp=\tw@ \else
2071     \l@dttempcntb=\z@ \l@dttempcnta=\@ne
2072     \fi
2073     \or
2074     \ifnum\sublock@disp=\z@
2075     \l@dttempcntb=\z@ \l@dttempcnta=\@ne
2076     \fi
2077 \fi}
2078 %

```

Similarly for line numbers.

```

2079 \newcommand*{\ch@ck@l@ck}{%
2080     \ifcase\@lock
2081     \or
2082     \ifnum\lock@disp=\@ne
2083     \l@dttempcntb=\z@ \l@dttempcnta=\@ne
2084     \fi
2085     \or
2086     \ifnum\lock@disp=\tw@ \else

```

```

2087     \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2088     \fi
2089     \or
2090     \ifnum\lock@disp=\z@
2091     \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2092     \fi
2093 \fi}
2094 %

```

Fix the lock counters. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

2095 \newcommand*\f@x@l@cks}{%
2096 \ifcase\@lock
2097 \or
2098 \global\@lock=\tw@
2099 \or \or
2100 \global\@lock=\z@
2101 \fi
2102 \ifcase\sub@lock
2103 \or
2104 \global\sub@lock=\tw@
2105 \or \or
2106 \global\sub@lock=\z@
2107 \fi}
2108
2109 %

```

## X Pstart number printing in side

In side, the printing of pstart number is running like the printing of line number. There is only some differences:

- The pstarts counter is upgrade in the `\pend` command. Consequently, the `\affixpstart@num` command has not to upgrade it, unlike the `\affixline@num` which upgrades the lines counter.
- To print the pstart number only at the beginning of a pstart, and not in every line, a boolean test is made. The `\pstartnum` boolean is set to TRUE at every `\pend`. It is tried in the `\leftpstartnum` and `\rightpstartnum` commands. After the try, it is set to FALSE.

```

\leftpstartnum10
\rightpstartnum11 \newif\ifsidepstartnum
\ifsidepstartnum12 \newcommand*\f@ffixpstart@num}{%
2113 \ifsidepstartnum
2114 \if@twocolumn
2115 \if@firstcolumn
2116 \gdef\l@dld@ta{\llap{\leftpstartnum}}}%

```

```

2117         \else
2118             \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2119         \fi
2120     \else
2121         \@l@tempcntb=\line@margin
2122         \ifnum\@l@tempcntb>\@ne
2123             \advance\@l@tempcntb \page@num
2124         \fi
2125         \ifodd\@l@tempcntb
2126             \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2127         \else
2128             \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2129         \fi
2130     \fi
2131 \fi
2132
2133 }
2134 %
2135
2136 \newif\ifpstartnum
2137 \pstartnumtrue
2138 \newcommand*\leftpstartnum){
2139     \ifpstartnum\thepstart
2140     \kern\linenumsep\fi
2141     \global\pstartnumfalse
2142 }
2143 \newcommand*\rightpstartnum){
2144     \ifpstartnum
2145     \kern\linenumsep
2146     \thepstart
2147     \fi
2148     \global\pstartnumfalse
2149 }
2150 %

```

## XI Restoring footnotes and penalties

Because of the paragraph decomposition process in order to number line, `reledmac` must hack the standard way  $\TeX$  works in order to manage insertion of footnotes, both critical and familiar.

We need to call the `\insert` commands not when the content of `\pstart... \pend` is read by  $\TeX$  but when each individual line is typeset.

Consequently, when reading the content of `\pstart... \pend`, we store the insertion (footnotes) in an specific `reledmac`'s list, and we restore them to the vertical list when printing each individual line.

## XI.1 Add insertions to the vertical list

`\inserts@list` `\inserts@list` is the list macro that contains the inserts that we save up for one paragraph.

```
2151 \list@create{\inserts@list}
2152 %
```

`\add@inserts` `\add@inserts` is the penultimate macro used by `\do@line`; it takes insertions saved in a list macro and sends them onto the vertical list.

It may call itself recursively, and to do this efficiently (using TeX's optimization for tail recursion), we define a control-sequence called `\add@inserts@next` that is always the last thing that `\add@inserts` does. If there could be more inserts to process for this line, `\add@inserts@next` is set equal to `\add@inserts`; otherwise it is just `\relax`.

```
2153 \newcommand*{\add@inserts}{%
2154   \global\let\add@inserts@next=\relax
2155 %
```

If `\inserts@list` is empty, there are not any more notes or insertions for this paragraph, and we need not waste our time.

```
2156   \ifx\inserts@list\empty \else
2157 %
```

The `\next@insert` macro records the number of the line that receives the next footnote or other insert; it is empty when we start out, and just after we have affixed a note or insert.

```
2158   \ifx\next@insert\empty
2159     \ifx\insertlines@list\empty
2160       \global\noteschanged@true
2161       \gdef\next@insert{100000}%
2162     \else
2163       \gl@p\insertlines@list\to\next@insert
2164     \fi
2165   \fi
2166 %
```

If the next insert's for this line, tack it on (and then erase the contents of the insert macro, as it could be quite large). In that case, we also set `\add@inserts@next` so that we will call ourselves recursively: there might be another insert for this same line.

```
2167   \ifnum\next@insert=\absline@num
2168     \gl@p\inserts@list\to\@insert
2169     \@insert
2170     \global\let\@insert=\undefined
2171     \global\let\next@insert=\empty
2172     \global\let\add@inserts@next=\add@inserts
2173   \fi
2174 \fi
2175 %
```

Make the recursive call, if necessary.

```
2176 \add@inserts@next}
2177
2178 %
```

## XI.2 Penalties

`\add@penalties` `\add@penalties` is the last macro used by `\do@line`. 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 this code, `\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` above (VIII p. 133). Finally, the penalty is checked to see that it does not go below  $-10000$ .

```
2179 \newcommand*{\add@penalties}{\@l@tempcnta=\ballast@count
2180   \ifnum\num@lines>\@ne
2181     \global\advance\par@line \@ne
2182     \ifnum\par@line=\@ne
2183       \advance\@l@tempcnta \clubpenalty
2184     \fi
2185     \@l@tempcntb=\par@line \advance\@l@tempcntb \@ne
2186     \ifnum\@l@tempcntb=\num@lines
2187       \advance\@l@tempcnta \widowpenalty
2188     \fi
2189     \ifnum\par@line<\num@lines
2190       \advance\@l@tempcnta \interlinepenalty
2191     \fi
2192   \fi
2193   \ifnum\@l@tempcnta=\z@
2194     \relax
2195   \else
2196     \ifnum\@l@tempcnta>-10000
2197       \penalty\@l@tempcnta
2198     \else
2199       \penalty -10000
2200     \fi
2201   \fi}
2202
2203 %
```

## XI.3 Printing leftover notes

`\flush@notes` The `\flush@notes` macro is called after the entire paragraph has been sliced up and sent on to the vertical list. If the number of notes to this paragraph has increased since

the previous run of  $\TeX$ , then there can be leftover notes that have not yet been printed. An appropriate error message will be printed elsewhere; but it is best to go ahead and print these notes somewhere, even if it is not in quite the right place. What we do is dump them all out here, so that they should be printed on the same page as the last line of the paragraph. We can hope that is not too far from the proper location, to which they will move on the next run.

```

2204 \newcommand*{\flush@notes}{%
2205   \@xloop
2206   \ifx\inserts@list\empty \else
2207     \glp\inserts@list\to\@insert
2208     \@insert
2209     \global\let\@insert=\undefined
2210   \repeat}
2211
2212 %

```

`\@xloop` `\@xloop` is a variant of the PLAIN  $\TeX$  `\loop` macro, useful when it's hard to construct a positive test using the  $\TeX$  `\if` commands—as in `\flush@notes` above. One types `\@xloop ... \if ... \else ... \repeat`, and the action following `\else` is repeated as long as the `\if` test fails. (This macro will work wherever the PLAIN  $\TeX$  `\loop` is used, too, so we could just call it `\loop`; but it seems preferable not to change the definitions of any of the standard macros.)

This variant of `\loop` was introduced by Alois Kabelašchacht in *TUGboat* **8** (1987), pp. 184–5.

```

2213 \def\@xloop#1\repeat{%
2214   \def\body{#1\expandafter\body\fi}%
2215   \body}
2216
2217 %

```

## XII Critical footnotes

The footnote macros are adapted from those in PLAIN  $\TeX$ , but they differ in these respects: the outer-level commands must add other commands to a list macro rather than doing insertions immediately; there are many separate levels of the footnotes, not just one; and there are options to reformat footnotes into paragraphs or into multiple columns.

### XII.1 Fonts

Before getting into the details of formatting the notes, we set up some font macros. It is the notes that present the greatest challenge for our font-handling mechanism, because we need to be able to take fragments of our main text and print them in different forms: it is common to reduce the size, for example, without otherwise changing the fonts used.

`\select@lemmafont` `\select@lemmafont` is provided to set the right font for the lemma in a note. This macro extracts the font specifier from the line and page number cluster, and issues the associated font-changing command, so that the lemma is printed in its original font.

```

2218 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@lemmafont#7|}
2219 \def\select@@lemmafont#1/#2/#3/#4|{%
2220   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2221   \selectfont}
2222
2223 %

```

## XII.2 Individual note options

`\footnoteoptions@` The `\footnoteoption@[side]{options}{value}` changes the value of on options of Xfootnote, to switch between true and false.

```

2224 \newcommand*\footnoteoptions@[3]{%
2225   \def\do##1{%
2226     \ifstrequal{#1}{L}{% In Leftside
2227       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
2228         unexpanded{##1}}}{\to\inserts@list%
2229         \global\advance\insert@count \@ne% Increment the left insert
2230         counter.
2231         }%
2232         }%
2233       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
2234         unexpanded{##1}}}{\to\inserts@listR%
2235         \global\advance\insert@countR \@ne% Increment the right insert
2236         counter insert.
2237         }%
2238         }%
2239       \notblank{#2}{\docsvlist{#2}}}{% Parsing all options
2240     }
2241   %

```

## XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua<sub>TeX</sub> is used.

```

2238 \newcommandx*\footnotelang@lua[1][1=L,usedefault]{%
2239   \ifstrequal{#1}{L}{%
2240     \xright@appenditem{\csxdef{footnote@luatextdir}{\the\textdir}}{\to\
2241     inserts@list%Know the dir of lemma
2242     \global\advance\insert@count \@ne%
2243     \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}{\to\
2244     inserts@list%Know the dir of lemma
2245     \global\advance\insert@count \@ne%
2246     }%

```



```

2245   {%
2246   \xright@appenditem{{\csxdef{footnote@luatextdir}{\the\textdir}}}\to\
inserts@listR%Know the dir of lemma
2247   \global\advance\insert@countR \@ne%
2248   \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\pardir}}}\to\
inserts@listR%Know the dir of lemma
2249   \global\advance\insert@countR \@ne%
2250   }%
2251 }
2252 %

```

`\footnotelang@poly` `\footnotelang@poly` is called to remember the information about the language of a lemma when polyglossia is used.

```

2253 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2254   \ifstrequal{#1}{L}{%
2255     \if@RTL%
2256       \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language used in the lemma
2257       \global\advance\insert@count \@ne%
2258     \else
2259       \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2260       \global\advance\insert@count \@ne%
2261     \fi%
2262     \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\languagename}}}\
to\inserts@listR%Know the language of lemma
2263     \global\advance\insert@count \@ne%
2264   }%
2265   {%
2266     \if@RTL
2267       \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
2268       \global\advance\insert@countR \@ne%
2269     \else
2270       \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2271       \global\advance\insert@countR \@ne%
2272     \fi
2273     \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\languagename}}}\
to\inserts@listR%Know the language of lemma
2274     \global\advance\insert@countR \@ne%
2275   }%
2276 }
2277 %

```

## XII.4 General survey of the way we manage notes

The processing of each note is done by four principal macros: the `\vfootnote` macro takes the text of the footnote and does the `\insert`; it calls on the `\footfmt` macro

to select the right fonts, print the line number and lemma, and do any other formatting needed for that individual note. Within the output routine, the two other macros, `\footstart` and `\footgroup`, are called; the first prints extra vertical space and a footnote rule, if desired; the second does any reformatting of the whole set of the footnotes in this series for this page—such as paragraphing or division into columns—and then sends them to the page.

These four macros, and the other macros and parameters shown here, are distinguished by the ‘series letter’ that indicates which set of the footnotes we are dealing with—A, B, C, D, or E. The series letter always precedes the string `foot` in macro and parameter names. Hence, for the A series, the four macros are called `\Afootnote`, `\Afootfmt`, `\Afootstart`, and `\Afootgroup`.

These macros are changed depending of the footnotes arrangement: “normal”, “paragraphed”, “two columns” or “three columns”.

## XII.5 General setup

`\footsplitskips` Some setup code that is common for a variety of the footnotes. The setup is for:

- `\interlinepenalty`.
- `\splittopskip` (skip before last part of notes that flow from one page to another).
- `\splitmaxdepth`.
- `\floatingpenalty`, that is penalty values being added when a long note flows from one page to another. Here, we let it to 0 when we are processing parallel pages in `eledpar`, in order to allow notes to flow from left to right pages and *vice-versa*. Otherwise, we let it to `\@MM`, which is the standard  $\TeX$  `\floatingpenalty`.

```

2278 \newcommand*\footsplitskips}{%
2279   \interlinepenalty=\interfootnotelinepenalty
2280   \unless\ifl@dprintingpages%
2281     \floatingpenalty=\@MM%
2282   \fi%
2283   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2284   \leftskip=\z@skip \rightskip=\z@skip}
2285
2286 %
```

`\normalfootnoterule` `\normalfootnoterule` is a standard footnote-rule macro, for use by a `footstart` macro: just the same as the PLAIN  $\TeX$  footnote rule.

```

2287 \let\normalfootnoterule=\footnoterule
2288 %
```

## XII.6 Footnotes arrangement

### XII.6.1 User level macro

`\Xarrangement` [*s*] {*arrangement*} The command calls, for each series, a specific command which set many counters and commands in order to define specific arrangement.

```

2289 \newcommand{\Xarrangement}[2][1,usedefault]{%
2290   \def\do##1{%
2291     \csname Xarrangement@#2\endcsname{##1}%
2292   }%
2293   \ifstrempy{#1}%
2294     {%
2295       \dolistloop{\@series}%
2296     }%
2297     {
2298       \docsvlist{#1}%
2299     }%
2300 }%
2301 %

```

### XII.6.2 Normal footnote

`\Xarrangement@normal` We can now define all the parameters for the series of footnotes; initially they use the “normal” footnote formatting.

What we want to do here is to insert something like the following for each footnote series. (This is an example, not part of the actual `reledmac` code.)

```

\skip\Afootins=12pt plus5pt minus5pt
\count\Afootins=1000
\dimen\Afootins=0.8\vszize
\let\Afootnote=\normalvfootnote \let\Afootfmt=\normalfootfmt
\let\Afootstart=\normalfootstart \let\Afootgroup=\normalfootgroup
\let\Afootnoterule=\normalfootnoterule

```

(Read *The TeXbook* in order to understand what are the counter, skip and dimen associated to an insertion.)

Instead of repeating ourselves, we define a `\Xarrangement@normal` macro that makes all these assignments for us, for any given series letter. This command is called when people use `\Xarrangement` [*series*] {normal}

Now we set up the `\Xarrangement@normal` macro itself. It takes one argument: the footnote series letter.

```

2302 \newcommand*{\Xarrangement@normal}[1]{%
2303   \csgdef{series@display#1}{normal}
2304   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2305   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2306   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt

```

```

2307 \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2308 \expandafter\let\csname #1footnoterule\endcsname=
2309 \normalfootnoterule
2310 \count\csname #1footins\endcsname=1000
2311 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2312 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2313 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2314 %

```

The `reledpar` provides tools in order to confine notes to one side. The mechanism is explained in the `reledpar`'s handbook. For now, just retain we need to store default value of the counter associated to the notes  $\TeX$ 's inserts.

```

2315 \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
2316 side only
2317 %

```

Now do the setup for minipage footnotes. We use as much as possible of the normal setup as we can (so the notes will have a similar layout).

```

2317 \ifnoledgroup@else%
2318 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2319 \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2320 \count\csname mp#1footins\endcsname=1000
2321 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2322 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2323 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2324 \fi
2325 }
2326
2327 %

```

`\normalvfootnote` We now begin a series of commands that do 'normal' footnote formatting: a format much like that implemented in PLAIN  $\TeX$ , in which each footnote is a separate paragraph.

`\normalvfootnote` takes the series letter as #1, and the entire text of the footnote is #2. It does the `\insert` for this note, calling on the `\footfmt` macro for this note series to format the text of the note.

```

2328 \notbool{parapparatus@}\newcommand*{\newcommand}{\normalvfootnote}[2]{%
2329 \insert\csname #1footins\endcsname\bgroup
2330 \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
2331 \noindent\csuse{Xbhooknote@#1}%
2332 \csuse{Xnotefontsize@#1}%
2333 \footplitskips
2334 \ifl@dpairing\ifl@dpaging\else%
2335 \setXnoteswidthliketwocolumns@#1}%
2336 \fi\fi%
2337 \setXnotespositionliketwocolumns@#1}%
2338 \spaceskip=\z@skip \xspaceskip=\z@skip
2339 \csname #1footfmt\endcsname #2{#1}\egroup}
2340 %

```

`\mpnormalvfootnote` And a somewhat different version for minipages.

```

2341 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2342   \global\setbox\@nameuse{mp#1footins}\vbox{%
2343     \unvbox\@nameuse{mp#1footins}
2344     \noindent\csuse{Xbhooknote@#1}%
2345     \csuse{Xnotefontsize@#1}%
2346     \hsize\columnwidth
2347     \@parboxrestore
2348     \color@begingroup
2349     \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2350
2351 %

```

`\normalfootfmt` `\normalfootfmt` is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 87), and the desired text, and output what’s to be printed. Argument #1 contains the line and page number information and lemma font specifier; #2 is the lemma; #3 is the note’s text. This version is very rudimentary—it uses `\printlines` to print just the range of line numbers, followed by a square bracket, the lemma, and the note text.

```

2352
2353
2354 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2355   \Xledsetnormalparstuff{#4}%
2356   \hangindent=\csuse{Xhangindent@#4}
2357   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2358   \strut{\printlinefootnote{#1}{#4}}%
2359   {\nottoggle{Xlemmadisablefontselection@#4}%
2360     {\select@lemmafnt#1|{\csuse{Xlemmafnt@#4}#2}}%
2361     {\csuse{Xlemmafnt@#4}#2}}%
2362   }%
2363   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2364     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2365     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2366     }}%
2367   #3\strut\par}
2368 %

```

`\normalfootstart` `\normalfootstart` is a standard footnote-starting macro, called in the output routine whenever there are footnotes of this series to be printed: it skips a bit and then draws a rule.

Any `\footstart` macro must put onto the page something that takes up space exactly equal to the `\skipXfootins` value for the associated series of notes.  $\TeX$  makes page computations based on that `\skip` value, and the output pages will suffer from spacing problems if what you add takes up a different amount of space.

But if the skip `\preXnotes@` is greater than 0 pt, it is used instead of `\skip\footins` for the first printed series in one page.

The `\leftskip` and `\rightskip` values are both zeroed here. Similarly, these skips are cancelled in the `\vfootnote` macros for the various types of notes. Strictly speaking, this is necessary only if you are using paragraphed footnotes, but we have put it here and in the other `\vfootnote` macros too so that the behavior of `reledmac` in this respect is general across all footnote types. What this means is that any `\leftskip` and `\rightskip` you specify applies to the main text, but not the footnotes. The footnotes continue to be of width `\hsize`.

```
2369 \newcommand*{\normalfootstart}[1]{%
2370 %
```

The first series of notes printed in a page can have a specific skip before it. In order to insert this specific skip without overlap the bottom margin of the page, Maïeul Rouquette have defined an algorithm explained in XVIII p. 196. Here is part of this algorithm, when the block of notes are ready to be printed.

```
2371 \ifdimequal{0pt}{\preXnotes@}{%
2372   {%
2373     \iftoggle{preXnotes@}{%
2374       \togglefalse{preXnotes@}%
2375       \skip\csname #1footins\endcsname=%
2376       \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2377     }%
2378   }%
2379 }%
2380 \vskip\skip\csname #1footins\endcsname%
2381 %
```

And now, the problem of left and right skip for notes. Especially when using one feature of `reledpar` which allows to have the footnotes horizontal size as the size of columns printed by `\Columns`. Read XV p. 194 for the general description of the problem.

```
2382 \leftskip0pt \rightskip0pt
2383 \ifl@dpairing\else%
2384   \hsize=\old@hsize%
2385 \fi%
2386 \setXnoteswidthliketwocolumns@{#1}%
2387 \setXnotespositionliketwocolumns@{#1}%
2388 %
```

And now, print the footnote's rule to finish the footnote's introduction.

```
2389 \print@Xfootnoterule{#1}%
2390 \noindent\leavevmode}
2391 %
```

`\normalfootgroup` `\normalfootgroup` is a standard footnote-grouping macro: it sends the contents of the footnote-insert box to the output page without alteration.

```

2392 \newcommand*{\normalfootgroup}[1]{%
2393   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2394   \csuse{Xbhookgroup@#1}%
2395   \unvbox\csname #1footins\endcsname%
2396   \hsize=\old@hsize%
2397   }%
2398
2399 %

```

`\mpnormalfootgroup` A somewhat different version for minipages. Note that, in this case, we do not make distinctions between the `\Xfootgroup` and `\Xfootstarts` macros.

```

2400 \unless\ifnoledgroup@
2401 \newcommand*{\mpnormalfootgroup}[1]{%
2402   \vskip\skip\@nameuse{mp#1footins}
2403   \ifl@dpairing\ifparledgroup%
2404     \leavevmode\marks\parledgroup@{begin}%
2405     \marks\parledgroup@series{#1}%
2406     \marks\parledgroup@type{Xfootnote}%
2407   \fi\fi\normalcolor%
2408   \ifparledgroup%
2409     \ifl@dpairing%
2410     \else%
2411       \setXnoteswidthliketwocolumns@{#1}%
2412       \setXnotespositionliketwocolumns@{#1}%
2413       \print@Xfootnoterule{#1}%%
2414     \fi%
2415   \else%
2416     \setXnoteswidthliketwocolumns@{#1}%
2417     \setXnotespositionliketwocolumns@{#1}%
2418     \print@Xfootnoterule{#1}%%
2419   \fi%
2420   \setlength{\parindent}{0pt}
2421   {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2422   \csuse{Xbhookgroup@#1}%
2423   \unvbox\csname mp#1footins\endcsname}}
2424 \fi
2425 %

```

### XII.6.3 Paraphrased footnotes

The `paraphrased-footnote` option reformats all the footnotes of one series for a page into a single paragraph; this is especially appropriate when the notes are numerous and brief. The code is based on *The TeXbook*, pp. 398–400, with alterations for our environment. This algorithm uses a considerable amount of save-stack space: a  $\TeX$  of ordinary size may not be able to handle more than about 100 notes of this kind on a page.

`\Xarrangement@paragraph` The `\Xarrangement@paragraph` macro sets up everything for one series of the footnotes so that they will be paraphrased; it takes the series letter as argument. We include

the setting of `\count\footins` to 1000 for the footnote series just in case user is switching to paragraphed footnotes after having columnar ones, since they change this value (see below).

The argument of `\Xarrangement@footparagraph` is the letter denoting the series of notes to be paragraphed.

```

2426 \newcommand*\Xarrangement@paragraph}[1]{%
2427   \csgdef{series@display#1}{paragraph}
2428   \expandafter\newcount\csname #1prevpage@num\endcsname
2429   \expandafter\let\csname #1footstart\endcsname=\parafootstart
2430   \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2431   \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2432   \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2433   \count\csname #1footins\endcsname=1000
2434   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2435   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2436   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2437   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2438   \para@footsetup{#1}
2439 %

```

And the extra setup for minipages.

```

2440 \ifnoledgroup@else
2441   \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2442   \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2443   \count\csname mp#1footins\endcsname=1000
2444   \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2445   \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2446   \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2447   \fi
2448 }
2449 %

```

`\footfudgefiddle` For paragraphed footnotes  $\TeX$  has to estimate the amount of space required. If it underestimates this then the notes may get too long and run off the bottom of the text block. `\footfudgefiddle` can be increased from its default 64 (say, to 70) to increase the estimate.

```

2450 \providecommand{\footfudgefiddle}{64}
2451 %

```

`\para@footsetup` `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hsize`. We assume that the proper value of `\baselineskip` for the footnotes (normally 9 pt) has been set already. The argument of the macro is again the note series letter.

Peter Wilson thinks that `\columnwidth` should be used here for  $\LaTeX$  not `\hsize`. Peter Wilson have also included `\footfudgefiddle`.



```

2452 \newcommand*{\para@footsetup}[1]{\csuse{Xhookgroup@#1}\csuse{
Xnotefontsize@#1}
2453 \setXnoteswidthliketwocolumns@{#1}%
2454 \ifcsemt{Xwidth@#1}%
2455   {}%
2456   {\columnwidth=\expandafter\dimexpr\csuse{Xwidth@#1}\relax}%
2457 \dimen0=\baselineskip
2458 \multiply\dimen0 by 1024
2459 \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2460 \csxdef{#1footfudgefactor}{%
2461   \expandafter\strip@pt\dimen0 }}}
2462
2463 %

```

`\strip@pt` strip the characters `pt` from a `dimen` value.

`\parafootstart` `\parafootstart` is the same as `\normalfootstart`, but we give it again to ensure that `\rightskip` and `\leftskip` are zeroed (this needs to be done before `\para@footgroup` in the output routine). The size of paragraphed notes is calculated using a fudge factor which in turn is based on `\hsize`. So the paragraph of notes needs to be that wide.

The argument of the macro is again the note series letter.

```

2464 \newcommand*{\parafootstart}[1]{%
2465 \rightskip=0pt \leftskip=0pt%
2466 \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
2467 \ifdimequal{0pt}{\preXnotes@}{}%
2468   {%
2469     \iftoggle{preXnotes@}{%
2470       \togglefalse{preXnotes@}%
2471       \skip\csname #1footins\endcsname=%
2472       \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2473     }%
2474   }%
2475 }%
2476 \vskip\skip\csname #1footins\endcsname%
2477 \setXnoteswidthliketwocolumns@{#1}%
2478 \setXnotespositionliketwocolumns@{#1}%
2479 \print@Xfootnoterule{#1}%
2480 \let\@bidi@RTL@everypar\@empty%
2481 \noindent\leavevmode}
2482 %

```

`\paravfootnote` `\paravfootnote` is a version of the `\vfootnote` command that is used for paragraphed notes. It gets appended to the `\inserts@list` list by an outer-level footnote command like `\Afootnote`. The first argument is the note series letter; the second is the full text of the printed note itself, including line numbers, lemmata, and footnote text.

The initial model for this insertion is, of course, the `\insert\footins` definition in *The TeXbook*, p. 398. There, the footnotes are first collected up in `hboxes`, and these `hboxes` are later unpacked and stuck together into a paragraph.

However, Michael Downes has pointed out that because text in `hboxes` gets typeset in restricted horizontal mode, there are some undesirable side-effects if you later want to break such text across lines. In restricted horizontal mode, where  $\TeX$  does not expect to have to break lines, it does not insert certain items like `\discretionary`s. If you later unbox these `hboxes` and stick them together, as the *TeXbook* macros do to make these footnotes, you lose the ability to hyphenate after an explicit hyphen. This can lead to overfull `hboxes` when you would not expect to find them, and to the uninitiated it might be very hard to see why the problem had arisen.<sup>27</sup>

Wayne Sullivan pointed out to us another subtle problem that arises from the same cause:  $\TeX$  also leaves the `\language` whatsit nodes out of the horizontal list.<sup>28</sup> So changes from one language to another will not invoke the proper hyphenation rules in such footnotes. Since critical editions often do deal with several languages, especially in a footnotes, we really ought to get this bit of code right.

To get around these problems, Wayne suggested emendations to the *TeXbook* versions of these macros which are broadly the same as those described by Michael: the central idea (also suggested by Donald Knuth in a letter to Michael) is to avoid collecting the text in an `hbox` in the first place, but instead to collect it in a `vbox` whose width is (virtually) infinite. The text is therefore typeset in unrestricted horizontal mode, as a paragraph consisting of a single long line. Later, there is an extra level of unboxing to be done: we have to unpack the `vbox`, as well as the `hboxes` inside it, but that is not too hard. For details, we refer you to Michael's article, where the issues are clearly explained.<sup>29</sup> Michael's unboxing macro is called `\Xunvxh`: `unvbox`, extract the last line, and `unhbox` it.

Doing things this way has an important consequence: as Michael pointed out, you really can't put an explicit line-break into a note built in a `vbox` the way we are doing.<sup>30</sup> In other words, be very careful not to use `\break`, or `\penalty-10000`, or any equivalent inside your para-footnote. If you do, most of the note will probably disappear. You *are* allowed to make strong suggestions; in fact `\penalty-9999` will be quite okay. Just do not make the break mandatory. We have not applied any of Michael's solutions here, since we feel that the problem is exiguous, and `reledmac` is quite baroque enough already. If you think you are having this problem, look up Michael's solutions.

One more thing; we set `\leftskip` and `\rightskip` to zero. This has the effect of neutralizing any such skips which may apply to the main text (cf. XII.6.2 p. 150 above). We need to do this, since `\footfudgefactor` is calculated on the assumption that the notes are `\hspace` wide.

So, finally, here is the modified foot-paragraph code, which sets the footnote in vertical mode so that language and discretionary nodes are included.

```
2483 \newcommand*{\paravfootnote}[2]{%
```

<sup>27</sup>Michael Downes, 'Line Breaking in `\unboxed` Text', *TUGboat* 11 (1990), pp. 605–612.

<sup>28</sup>See *The TeXbook*, p. 455 (editions after January 1990).

<sup>29</sup>Wayne supplied his own macros to do this, but since they were almost identical to Michael's, Peter Wilson has used the latter's `\Xunvxh` macro since it is publicly documented.

<sup>30</sup>'Line Breaking', p. 610.

```

2484 \insert\csname #1footins\endcsname
2485 \bgroup
2486 \csuse{Xnotefontsize@#1}
2487 \footsplitskips
2488 \setbox0=\vbox{\hsize=\maxdimen%
2489 \let\bidir@RTL@everypar\@empty%
2490 \noindent\csuse{Xhooknote@#1}%
2491 \csname #1footfmt\endcsname #2{#1}}%
2492 \setbox0=\hbox{\Xunvxh{0}{#1}}%
2493 \dp0=0pt
2494 \ht0=\csname #1footfudgefactor\endcsname\wd0
2495 %

```

Here we produce the contents of the footnote from box 0, and add a penalty of 0 between boxes in this insert.

```

2496 \if@RTL\noindent \leavevmode\fi\box0%
2497 \penalty0
2498 \egroup}
2499 %
2500 %

```

The final penalty of 0 was added here at Wayne's suggestion to avoid a weird page-breaking problem, which occurs on those occasions when  $\TeX$  attempts to split foot paragraphs. After trying out such a split (see *The TeXbook*, p. 124),  $\TeX$  inserts a penalty of  $-10000$  here, which nearly always forces the break at the end of the whole footnote paragraph (since individual notes can't be split) even when this leads to an overfull vbox. The change above results in a penalty of 0 instead which allows, but does not force, such breaks. This penalty of 0 is later removed, after page breaks have been decided, by the `\unpenalty` macro in `\makehboxofhboxes`. So it does not affect how the footnote paragraphs are typeset (the notes still have a penalty of  $-10$  between them, which is added by `\parafootfmt`).

`\mpparavfootnote` This version is for minipages.

```

2501 \newcommand*\mpparavfootnote}[2]{%
2502 \global\setbox\@nameuse{mp#1footins}\vbox{%
2503 \unvbox\@nameuse{mp#1footins}%
2504 \csuse{Xnotefontsize@#1}
2505 \footsplitskips
2506 \setbox0=\vbox{\hsize=\maxdimen%
2507 \let\bidir@RTL@everypar\@empty%
2508 \noindent\color@begingroup%
2509 \csuse{Xhooknote@#1}%
2510 \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2511 \setbox0=\hbox{\Xunvxh{0}{#1}}%
2512 \dp0=\z@
2513 \ht0=\csname #1footfudgefactor\endcsname\wd0
2514 \box0
2515 \penalty0
2516 }}

```

```
2517
2518 %
```

`\Xunvxh` Here is (modified) Michael’s definition of `\unvxh`, used above. Michael’s macro also takes care to remove some unwanted penalties and glue that T<sub>E</sub>X automatically attaches to the end of paragraphs. When T<sub>E</sub>X finishes a paragraph, it throws away any remaining glue, and then tacks on the following items: a `\penalty` of 10000, a `\parfillskip` and a `\rightskip` (*The TeXbook*, pp. 99–100). `\unvxh` cancels these unwanted paragraph-final items using `\unskip` and `\unpenalty`.

```
2519 \newcommand*\Xunvxh}[2]{%
2520   \setbox0=\vbox{\unvbox#1%
2521     \global\setbox1=\lastbox}%
2522   \unhbox1
2523   \unskip           % remove \rightskip,
2524   \unskip           % remove \parfillskip,
2525   \unpenalty       % remove \penalty of 10000,
2526   \hskip\csuse{Xafternote@#2}} % but add the glue to go between the notes
2527
2528 %
```

`\parafootfmt` `\parafootfmt` is `\normalfootfmt` adapted to do the special stuff needed for paragraphed notes—leaving out the `\endgraf` at the end, sticking in special penalties and kern, and leaving out the `\footstrut`. The first argument is the line and page number information, the second is the lemma, the third is the text of the footnote, and the fourth is the series (optional, for backward compatibility).

```
2529 \newcommand*\parafootfmt}[4]{%
2530   \Xinsertparafootsep{#4}%
2531   \ledsetnormalparstuff@common%
2532   \printlinefootnote{#1}{#4}%
2533   {\nottoggle{Xlemmadisablefontselection@#4}%
2534     {\select@lemmafонт#1|{\csuse{Xlemmafонт@#4}#2}}}%
2535     {\csuse{Xlemmafонт@#4}#2}}%
2536   }%
2537   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
Xlemmaseparator@#4}%
2538     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2539     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2540   }}%
2541   #3\penalty-10 }
2542 %
```

Note that in the above definition, the penalty of  $-10$  encourages a line break between notes, so that notes have a slight tendency to begin on new lines. The `\Xinsertparafootsep` command is used to insert the `\Xparafootsep@series` between each note in the *same* page.

`\parafootgroup` This footgroup code is modelled on the macros in *The TeXbook*, p. 399. The only difference is the `\unpenalty` in `\makehboxofhboxes`, which is there to remove the penalty of 0 which was added to the end of each footnote by `\para@vfootnote`.

The call to `\Xnotefontsize@{s}` is to ensure that the correct `\baselineskip` for the footnotes is used. The argument is the note series letter.

```

2543 \newcommand*{\parafootgroup}[1]{%
2544   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
2545   \unvbox\csname #1footins\endcsname
2546   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2547   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2548   \makehboxofhboxes
2549   \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2550   \csuse{Xhookgroup@#1}%
2551   \csuse{Xnotefontsize@#1}%
2552   \unhbox0\par%
2553   \global\hsize=\old@hsize%
2554   }%
2555
2556 %

```

`\mpparafootgroup` The minipage version.

```

2557 \newcommand*{\mpparafootgroup}[1]{%
2558   \setXnoteswidthliketwocolumns@{#1}%
2559   \vskip\skip\@nameuse{mp#1footins}
2560   \ifl@dpairing\ifparledgroup%
2561     \leavevmode\marks\parledgroup@{begin}%
2562     \marks\parledgroup@series{#1}%
2563     \marks\parledgroup@type{Xfootnote}%
2564   \fi\fi\normalcolor
2565   \ifparledgroup%
2566     \ifl@dpairing%
2567     \else%
2568       \setXnoteswidthliketwocolumns@{#1}%
2569       \setXnotespositionliketwocolumns@{#1}%
2570       \print@Xfootnoterule{#1}%%
2571     \fi%
2572   \else%
2573     \setXnoteswidthliketwocolumns@{#1}%
2574     \setXnotespositionliketwocolumns@{#1}%
2575     \print@Xfootnoterule{#1}%
2576   \fi%
2577   \unvbox\csname mp#1footins\endcsname
2578   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2579   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2580   \makehboxofhboxes
2581   \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%

```

```

2582 \csuse{Xhookgroup@#1}%
2583 \csuse{Xnotefontsize@#1}%
2584 \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
2585 \unhbox0\par}}
2586
2587 %

```

And finally, the two macros which are required to transform the long horizontal box stored in the insert' box to a printable text.

```

\makehboxofhboxes 2588 \newcommand*\makehboxofhboxes{\setbox0=\hbox{}}%
\removehboxes 2589 \loop
2590 \unpenalty
2591 \setbox2=\lastbox
2592 \ifhbox2
2593 \setbox0=\hbox{\box2\unhbox0}%
2594 \repeat}
2595
2596 \newcommand*\removehboxes{\setbox0=\lastbox
2597 \ifhbox0{\removehboxes}\unhbox0 \fi}
2598
2599 %

```

**Insertion of the footnotes separator** The command `\Xinsertparafootsep{<series>}` must be called at the beginning of `\parafootftm`.

```

\prevpage@num 2600 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep 2601 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
2602 {\ifcndef{prevline#1}% Be sur \prevline#1 exists.
2603 {\ifnumequal{\csuse{prevline#1}}{\line@num}%
2604 {\ifcempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
2605 {\csuse{Xparafootsep@#1}}%
2606 }%
2607 {\csuse{Xparafootsep@#1}}%
2608 }%
2609 {}%
2610 \global\csname #1prevpage@num\endcsname=\page@num%
2611 }
2612 %

```

## XII.6.4 Columnar footnotes

### Common tools

```

\rigidbalance
\rigidbalanceX
\Xrigidbalance
\dosplits
\splitoff
\@h
\@k

```

We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The

`\rigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the syntax of the arguments so that they do not depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

The  $\TeX$  `\line` macro has no relationship to the TeX `\line`. The  $\TeX$  equivalent is `\@@line`.

We do not call directly `\rigidbalance`, but we call `\Xrigidbalance` for critical notes and `\rigidbalanceX` for familiar notes. Both of them call `\rigidbalance`.

```

2613 \newcount\@k \newdimen\@h
2614 \newcommand*\Xrigidbalance}[3]{%
2615   \hsize=\expandafter\dimexpr\csuse{Xwidth@\@currentseries}\relax%
2616   \rigidbalance{#1}{#2}{#3}%
2617 }%
2618
2619 \newcommand*\rigidbalanceX}[3]{%
2620   \hsize=\expandafter\dimexpr\csuse{widthX@\@currentseries}\relax%
2621   \rigidbalance{#1}{#2}{#3}%
2622 }%
2623
2624 \newcommand*\rigidbalance}[3]{%
2625   \setbox0=\box#1 \@k=#2 \@h=#3%
2626   \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
2627   \valign{##\vfil\cr\dosplits}}
2628
2629 \newcommand*\dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
2630   \global\advance\@k-1\cr\dosplits\fi}
2631
2632 \newcommand*\splitoff{\dimen0=\ht0
2633   \divide\dimen0 by\@k \advance\dimen0 by\@h
2634   \setbox2 \vsplit0 to \dimen0
2635   \unvbox2 }
2636
2637 %

```

### Three columns

```

\Xarrangement@threecol 2638 \newcommand*\Xarrangement@threecol}[1]{%
2639   \csgdef{series@display#1}{threecol}
2640   \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
2641   \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2642   \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2643   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2644   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2645   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2646   \threecolfootsetup{#1}
2647 %

```

The additional setup for minipages.

```

2648 \ifnoledgroup@else
2649 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2650 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
2651 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2652 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2653 \mpthreecolfootsetup{#1}
2654 \fi
2655 }
2656
2657 %

```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 149 above).

`\threecolfootsetup` The `\threecolfootsetup` macro calculates and sets some numbers for three-column footnotes.

We set the `\count` of the foot insert to 333. Each footnote can be thought of as contributing only one third of its height to the page, since the footnote insertion has been made as a long narrow column, which then gets trisected by the `\rigidbalance` routine (inside `\threecolfootgroup`). These new, shorter columns are saved in a box, and then that box is *put back* into the footnote insert, replacing the original collection of the footnotes. This new box is, therefore, only about a third of the height of the original one.

The `\dimen` value for this note series has to change in the inverse way: it needs to be three times the actual limit on the amount of space these notes are allowed to fill on the page, because when `TEX` is accumulating material for the page and checking that limit, it does not apply the `\count` scaling.

```

2658 \newcommand*{\threecolfootsetup}[1]{%
2659 \count\csname #1footins\endcsname 333
2660 \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2661 \multiply\dimen\csname #1footins\endcsname \thr@@}
2662 %

```

`\mpthreecolfootsetup` The setup for minipages.

```

2663 \newcommand*{\mpthreecolfootsetup}[1]{%
2664 \count\csname mp#1footins\endcsname 333
2665 \multiply\dimen\csname mp#1footins\endcsname \thr@@}
2666
2667 %

```

`\threecolvfootnote` `\threecolvfootnote` is the `\vfootnote` command for three-column notes. The call to `\Xnotefontsize@<s>` ensures that the `\splittopskip` and `\splitmaxdepth` take their values from the right `\strutbox`: the one used in a footnotes. Note especially the importance of temporarily reducing the `\hspace` to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal `\hspace` is, say, 10 cm,



then each column will be  $0.3 \times 10 = 3$  cm wide, leaving a gap of 1 cm spread equally between columns (i.e., .5 cm between each).

The arguments are #1 the note series letter and #1 the full text of the note (including numbers, lemma and text).

```

2668 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnote}[2]{%
2669   \insert\csname #1footins\endcsname\bgroup%
2670   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
2671   \noindent\csuse{Xhooknote@#1}%
2672   \csuse{Xnotefontsize@#1}%
2673   \footplitskips%
2674   \csname #1footfmt\endcsname #2{#1}\egroup}
2675 %

```

`\threecolfootfmt` `\threecolfootfmt` is the command that formats one note. The arguments are #1 the line numbers, #2 the lemma and #4 the text of the `-footnote` command #4 optional (for backward compatibility): the series.

```

2676 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmt}[4]{%
2677   \normal@pars%
2678   \hsize \csuse{Xsizethreecol@#4}%
2679   \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2680   \tolerance=5000%
2681   \hangindent=\csuse{Xhangindent@#4}%
2682   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2683   \@tempdima=\parindent%
2684   \csuse{Xcolalign@#4}%
2685   \parindent=\@tempdima%
2686   \strut{%
2687     \hspace{\parindent}%
2688     \printlinefootnote{#1}{#4}%
2689   }%
2690   {\nottoggle{Xlemmadisablefontselection@#4}%
2691     {\select@lemmafnt#1|{\csuse{Xlemmafnt@#4}#2}}%
2692     {\csuse{Xlemmafnt@#4}#2}}%
2693   }%
2694   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2695     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2696     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2697   }}%
2698   #3\strut\par\allowbreak}
2699 %

```

`\threecolfootgroup` And here is the `footgroup` macro that is called within the output routine to regroup the notes into three columns. Once again, the call to `\Xnotefontsize@(s)` is there to ensure that it is the right `\splittopskip`—the one used in footnotes—which is used to provide the third argument for `\rigidbalance`. This third argument (`\@h`) is the `topskip` for

the box containing the text of the footnotes, and does the job of making sure the top lines of the columns line up horizontally. In *The TeXbook*, p. 398, Donald Knuth suggests retrieving the output of `\rigidbalance`, putting it back into the insertion box, and then printing the box. Here, we just print the `\line` which comes out of `\rigidbalance` directly, without any re-boxing.

```

2700 \newcommand*\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2701 \noindent\csuse{Xtxtbeforenotes@#1}}%
2702 \csuse{Xhookgroup@#1}\par%
2703 \splittopskip=\ht\strutbox
2704 \expandafter
2705 \Xrigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
2706 %

```

`\mpthreecolfootgroup` The setup for minipages.

```

2707 \newcommand*\mpthreecolfootgroup}[1]{\%
2708 \vskip\skip\@nameuse{mp#1footins}
2709 \ifl@dpairing\ifparledgroup%
2710 \leavevmode\marks\parledgroup@{begin}%
2711 \marks\parledgroup@series{#1}%
2712 \marks\parledgroup@type{Xfootnote}%
2713 \fi\fi\normalcolor
2714 \ifparledgroup%
2715 \ifl@dpairing%
2716 \else%
2717 \setXnoteswidthliketwocolumns@{#1}%
2718 \setXnotespositionliketwocolumns@{#1}%
2719 \print@Xfootnoterule{#1}%
2720 \fi%
2721 \else%
2722 \setXnoteswidthliketwocolumns@{#1}%
2723 \setXnotespositionliketwocolumns@{#1}%
2724 \print@Xfootnoterule{#1}%
2725 \fi%
2726 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2727 \csuse{Xhookgroup@#1}\par%
2728 \splittopskip=\ht\strutbox
2729 \expandafter
2730 \Xrigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2731
2732 %

```

## Two columns

```

\Xarrangement@twocol33 \newcommand*\Xarrangement@twocol}[1]{\%
2734 \csgdef{series@display#1}{twocol}
2735 \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2736 \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt

```

```

2737 \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2738 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2739 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2740 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2741 \twocolfootsetup{#1}
2742 %

```

The additional setup for minipages.

```

2743 \ifnoledgroup@ \else
2744   \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2745   \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2746   \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2747   \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2748   \mptwocolfootsetup{#1}
2749 \fi
2750 }
2751
2752 %

```

`\twocolfootsetup` Here is a series of macros which are very similar to their three-column counterparts. In  
`\twocolvfootnote` this case, each note is assumed to contribute only a half a line of text. And the notes are  
`\twocolfootfmt` set in columns giving a gap between them of one tenth of the `\hsiz`.  
`\twocolfootgroup`

```

2753 \newcommand*{\twocolfootsetup}[1]{%
2754   \count\csname #1footins\endcsname 500
2755   \csxdef{default@#1footins}{500}%
2756   \multiply\dimen\csname #1footins\endcsname \tw@}
2757 %

2758 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolvfootnote}[2]{%
2759   \insert\csname #1footins\endcsname\bgroup%
2760   \hsize=\expandafter\dimexpr\csuse{Xwidth@#1}\relax%
2761   \noindent\csuse{Xhooknote@#1}%
2762   \csuse{Xnotefontsize@#1}%
2763   \footssplitsskips%
2764   \csname #1footfmt\endcsname #2{#1}\egroup}
2765 %

2766 \notbool{parapparatus@}{\newcommand*{\newcommand}{\twocolfootfmt}[4]{% 4th
  arg is optional, for backward compatibility
2767   \normal@pars%
2768   \hsize \csuse{Xhsizetwocol@#4}%
2769   \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2770   \tolerance=5000%
2771   \hangindent=\csuse{Xhangindent@#4}%
2772   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2773   \@tempdima=\parindent%
2774   \csuse{Xcolalign@#4}%
2775   \parindent=\@tempdima%
2776   \strut{}}

```

```

2777 \hspace{\parindent}%
2778 \printlinefootnote{#1}{#4}%
2779 }%
2780 {\nottoggle{Xlemmadisablefontselection@#4}%
2781  {\select@lemmafонт#1|{\csuse{Xlemmafонт@#4}#2}}%
2782  {\csuse{Xlemmafонт@#4}#2}}%
2783 }%
2784 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
Xlemmaseparator@#4}%
2785  {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2786  {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2787 }}%
2788 #3\strut\par\allowbreak}
2789 %

2790 \newcommand*{\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2791  \noindent\csuse{Xtxtbeforenotes@#1}}%
2792  \csuse{Xhookgroup@#1}\par%
2793  \splittopskip=\ht\strutbox
2794  \expandafter
2795  \Xrigidbalance\csname #1footins\endcsname \tw@ \splittopskip}
2796
2797 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

2798 \newcommand*{\mptwocolfootsetup}[1]{%
2799  \count\csname mp#1footins\endcsname 500
2800  \multiply\dimen\csname mp#1footins\endcsname \tw@}
2801 %

2802 \newcommand*{\mptwocolfootgroup}[1]{%
2803  \vskip\skip@nameuse{mp#1footins}
2804  \ifl@dpairing\ifparledgroup%
2805  \leavevmode\marks\parledgroup@{begin}%
2806  \marks\parledgroup@series{#1}%
2807  \marks\parledgroup@type{Xfootnote}%
2808  \fi\fi\normalcolor
2809  \ifparledgroup%
2810  \ifl@dpairing%
2811  \else%
2812  \setXnoteswidthliketwocolumns@{#1}%
2813  \setXnotespositionliketwocolumns@{#1}%
2814  \print@Xfootnoterule{#1}%
2815  \fi%
2816  \else%
2817  \setXnoteswidthliketwocolumns@{#1}%
2818  \setXnotespositionliketwocolumns@{#1}%
2819  \print@Xfootnoterule{#1}%

```

```

2820 \fi%
2821 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2822 \csuse{Xbhookgroup@#1}\par%
2823 \splittopskip=\ht\strutbox
2824 \expandafter
2825 \Xrigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2826
2827 %

```

## XII.7 Critical notes presentation

Here, we define some commons macro which are used in order to print a critical notes, that is a note with 1) line number 2) lemma 3) lemma separator 4) text associated to the lemma.

### XII.7.1 Font tools

`\endashchar` The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

The `\endashchar` macro is simply an en-dash from the normal font and is immune to changes in the surrounding font. The same goes for the full stop. These two are used in `\printlines`. The right bracket macro is the same again; it crops up in `\normalfootfmt` and the other footnote macros for controlling the format of the footnotes.

With `polyglossia`, each critical note has a `\footnote@lang` which shows the language of the lemma, and which can be used to switch the bracket from right to left.

```

2828 \def\endashchar{\textnormal{--}}
2829
2830 \newcommand*{\fullstop}{\textnormal{.}}
2831 \def\Xsublinesep@side{\fullstop}
2832
2833 \newcommand*{\rbracket}{\textnormal{%
2834   \csuse{text\csuse{footnote@lang}}}%
2835   \ifluatex%
2836     \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]{\thinspace
2837     ]}}%
2838     \else%
2839     \thinspace]}%
2840   \fi}%
2841 }
2842
2843 %

```

### XII.7.2 Pstart number in footnote

`\printpstart` The `\printpstart` macro prints the pstart number for a note.

```

2844 \newcommand{\printpstart}[0]{%
2845   \ifboolexpr{bool{!@dpairing} or bool{!@dprintingpages} or bool{
l@dprintingcolumns}}{%
2846     \ifledRcol%
2847       \thepstartR%
2848     \else%
2849       \thepstartL%
2850     \fi%
2851   }{%
2852     \thepstart%
2853   }%
2854 }
2855 %

```

### XII.7.3 Line number printing

`\printlinefootnote` The `\printlinefootnote` macro is called in each `\<type>footfmt` command. It controls whether the line number is printed or not, according to the series options. Its first argument is the information about lines; its second is the series of the footnote. The printing of the line number is shared in `\printlinefootnotenumbers`.

```

2856 \newcommand{\printlinefootnote}[2]{%
2857   \l@dp@rsefootspec#1|{%
2858   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
2859     \edef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub - \
l@dparsedendline - \l@dparsedendsub}%
2860   }%
2861   {%
2862     \edef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub}%
2863   }%
2864   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2865     \hspace{\csuse{Xinplaceofnumber@#2}}%
2866   }%
2867   {%
2868     {%
2869       \iftoggle{Xnonumber@#2}{%Try if the line number must printed (by
default, yes)
2870       {%
2871         \hspace{\csuse{Xinplaceofnumber@#2}}%
2872       }%
2873       {%
2874         {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2875         {%
2876           \ifcsdef{prevline#2}%

```

```

2877     {%Be sure the \prevline exists.
2878     \ifcsequal{prevline#2}{\lineinfo@}%Try it
2879     {%
2880     \ifcsequal{Xsymlinenum@#2}%
2881     {%
2882     \hspace{\csuse{Xinplaceofnumber@#2}}%
2883     }%
2884     {\printsymlinefootnotearea{#2}}%
2885     }%
2886     {%
2887     \printlinefootnotearea{#1}{#2}%
2888     }%
2889     }%
2890     {%
2891     \printlinefootnotearea{#1}{#2}%
2892     }%
2893     }%
2894     {%
2895     \printlinefootnotearea{#1}{#2}%
2896     }%
2897     \csxdef{prevline#2}{\lineinfo@}%
2898     }%
2899   }%
2900 }%
2901 }%
2902 }
2903 %

```

`\printsymlinefootnotearea` This macro prints the space before the line symbol, changes the font, when prints the line symbol and the space after it.

```

2904 \newcommand{\printsymlinefootnotearea}[1]{%
2905   \hspace{\csuse{Xbeforesymlinenum@#1}}%
2906   \csuse{Xnotenumfont@#1}%
2907   \ifdimequal{\csuse{Xboxsymlinenum@#1}}{\z@}%
2908     {\csuse{Xsymlinenum@#1}}%
2909     {\hbox to \csuse{Xboxsymlinenum@#1}%
2910       {\csuse{Xsymlinenum@#1}\hfill}}%
2911   }%
2912   \hspace{\csuse{Xaftersymlinenum@#1}}%
2913 }%
2914 %

```

`\printlinefootnotearea` This macro prints the space before the line number, changes the font, then prints the line number and the space after it. It is called by `\printlinefootnote` depending of the options about repeating line numbers. The first argument is line information, the second is the notes series (A, B, C, etc.)

```

2915 \newcommand{\printlinefootnotearea}[2]{%
2916   \printXbeforenumber{#2}%

```

```

2917 \csuse{Xnotenumfont@#2}%
2918 \boxfootnotenumbers{#1}{#2}%
2919 \printXafternumber{#2}%
2920 }%
2921 %

```

**\boxfootnotenumbers** Depending on the user settings, this macro will box line numbers (or not). The first argument is line information, the second is the notes series (A, B, C, etc.) The previous `\printlinefootnotearea` calls it.

```

2922 \newcommand{\boxfootnotenumbers}[2]{%
2923   \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
2924     \printlinefootnotenumbers{#1}{#2}%
2925   }%
2926   {%
2927     \hbox to \csuse{Xboxlinenum@#2}%
2928       {%
2929         \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2930         \printlinefootnotenumbers{#1}{#2}%
2931         \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2932       }%
2933     }%
2934   }%
2935   %

```

**\printlinefootnotenumbers** This macro prints, if needed, the pstart number and the line number. The first argument is line information, the second is the notes series (A, B, C, etc.) The previous `\boxlinefootnote` calls it.

```

2936 \newcommand{\printlinefootnotenumbers}[2]{%
2937   \xdef\@currentseries{#2}%
2938   \ifboolexpr{%
2939     (togl{Xpstart@#2} and bool{numberpstart})%
2940     or togl{Xpstarteverytime@#2}}%
2941     {\printpstart}{}%
2942     \iftoggle{Xstanza@#2}{%
2943       \ifnumberstanza%
2944         \printstanza%
2945         \csuse{Xstanzaseparator@#2}%
2946       \fi%
2947     }{%
2948       \iftoggle{Xonlypstart@#2}{%
2949         \csuse{Xtxtbeforenumber@#2}%
2950         \printlines#1||\ifledRcol@{\@Rlineflag}\fi}%
2951       }%
2952     %

```

**\printXbeforenumber** This macro prints a space (before the line number) in footnote. It is called by `\printlinefootnotearea`. Its only argument is the note series (A, B, C, etc.)



```

2953 \newcommand{\printXbeforenumber}[1]{%
2954   \hspace{\csuse{Xbeforenumber@#1}}%
2955 }%
2956 %

```

`\printXafternumber` This macro prints the space, adding eventually a `\nobreak`, after the line number, in footnote. It is called by `\printlinefootnotearea`. Its only argument is the series

```

2957 \newcommand{\printXafternumber}[1]{%
2958   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%
2959   \hspace{\csuse{Xafternumber@#1}}%
2960 }%
2961 %

```

If we have decided to print the line number in a specific notes, the `\printlines` macro prints the line numbers for a note—which, in the general case, is a rather complicated task. The seven parameters of the argument are the line numbers as stored in `\l@d@nums`, in the form described on V.9 p. 87: the starting page, line, and sub-line numbers, followed by the ending page, line, and sub-line numbers, and then the font specifier for the lemma.

edmac’ creator have defined six boolean in order to know which component of line number description we have to print:

- `\ifl@d@pnum` for page numbers;
- `\ifl@d@ssub` for starting sub-line;
- `\ifl@d@elin` for ending line;
- `\ifl@d@esl` for ending sub-line; and
- `\ifl@d@dash` for the dash between the starting and ending groups.

There is no boolean for the line number because it is always printed.

Maïeul Rouquette has added `\ifl@d@Xtwolines` and `\ifl@d@Xmorethantwolines` to print a symbol which stands for “and subsequent” when there are two, three or more lines.

```

\ifl@d@pnum62 \newif\ifl@d@pnum
\ifl@d@ssub63 \newif\ifl@d@ssub
\ifl@d@elin64 \newif\ifl@d@elin
\ifl@d@esl65 \newif\ifl@d@esl
\ifl@d@dash66 \newif\ifl@d@dash
\ifl@d@Xtwolines2967 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines2968 \newif\ifl@d@Xmorethantwolines%
2969 %

```

```

\l@dp@rsefootspec \l@dp@rsefootspec parses lines specification and defines macros which hold the nu-
\l@dparsedstartpage umeric values. Just a reminder of the arguments:
\l@dparsedstartline \printlines #1 | #2 | #3 | #4 | #5 | #6 | #7
\l@dparsedstartsub \printlines start-page | line | subline | end-page | line | subline | fontflag
\l@dparsedendpage
\l@dparsedendline
\l@dparsedendsub
}
%
```

Initialise the several number value macros.

```

\def\l@dparsedstartpage{0}%
\def\l@dparsedstartline{0}%
\def\l@dparsedstartsub{0}%
\def\l@dparsedendpage{0}%
\def\l@dparsedendline{0}%
\def\l@dparsedendsub{0}%
%
```

**\setprintlines** The macro `\setprintlines` does the work of deciding what numbers should be printed. Its arguments are the same as the first 6 of `\printlines`.

```

\newcommand*\setprintlines[6]{%
  \l@d@pnumfalse \l@d@dashfalse
%
```

We print the page numbers only if: 1) we are doing the lineation by page, and 2) the ending page number is different from the starting page number.a

```

\ifbypage@
  \ifnum#4=#1 \else
    \l@d@pnumtrue
    \l@d@dashtrue
  \fi
\fi
%
```

We print the ending line number if: (1) we are printing the ending page number, or (2) it is different from the starting line number.

```

\ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
\ifnum#2=#5 \else
  \l@d@elintrue
  \l@d@dashtrue
\fi
%
```

We print the starting sub-line if it is nonzero.

```

3003 \l@d@ssubfalse
3004 \ifnum#3=0 \else
3005     \l@d@ssubtrue
3006 \fi
3007 %

```

We print the ending sub-line if it is nonzero and: (1) it is different from the starting sub-line number, or (2) the ending line number is being printed.

```

3008 \l@d@eslfalse
3009 \ifnum#6=0 \else
3010     \ifnum#6=#3
3011         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3012     \else
3013         \l@d@esltrue
3014         \l@d@dashtrue
3015     \fi
3016 \fi%
3017 %

```

However, if the `\Xtwolines` is set for the current series, we do not print the last line number.

```

3018 \ifl@d@dash%
3019     \ifboolexpr{togl{fulllines@} or test{\ifcsemtyp{Xtwolines@}
3020 \currentseries}}}%
3021     {%
3022     \setistwofollowinglines{#1}{#2}{#4}{#5}%
3023     \ifboolexpr{%
3024         (%
3025             togl {Xtwolinesbutnotmore@\currentseries}%
3026             and not%
3027             (%
3028                 bool {istwofollowinglines@}%
3029             )%
3030         )%
3031     or%
3032     (%
3033         (not test{\ifnumequal{#1}{#4}})%
3034         and togl{Xtwolinesonlyinsamepage@\currentseries}%
3035     )%
3036     }%
3037     {%
3038     \l@d@dashfalse%
3039     \l@d@Xtwolinestrue%
3040     \l@d@elinfalse%
3041     \l@d@eslfalse%
3042     \ifcsemtyp{Xmorethantwolines@\currentseries}%
3043

```

```

3044     {}%
3045     {\ifistwofollowinglines@else%
3046       \l@d@Xmorethantwolinestrue%
3047     \fi%
3048   }%
3049 }%
3050 }%
3051 \fi%
3052 %

```

End of `\setprintlines`.

```

3053 }%
3054 %

```

`\setistwofollowinglines` The `\ifistwofollowinglines` boolean, used by the `\Xtwolines` and related setting, is set to true by `\setistwofollowinglines`. This command takes the following arguments:

- #1 First page number.
- #2 First line number.
- #3 Last page number.
- #4 Last line number.

If  $\#3 - \#2 = 1$ , then that means the two lines are subsequent, and consequently `\ifistwofollowinglines` is set to true. However, if we use lineation by page, two given lines can be subsequent if:

- The first line number is equal to the last line number of the first page.
- The last line number is equal to 1.
- $\#3 - \#1$  is equal to 1.

```

3055 \newif\ifistwofollowinglines@%
3056 \newcommand{\setistwofollowinglines}[4]{%
3057   \ifcsdef{lastlinenumberon#1}%
3058     {\numdef{\tmp}{\csuse{lastlinenumberon#1}}}%
3059     {\numdef{\tmp}{0}}%
3060   \istwofollowinglines@false%
3061   \ifnumequal{#4-#2}{1}%
3062     {\istwofollowinglines@true}%
3063   {\ifbypage@%
3064     \ifnumequal{#3-#1}{1}%
3065     {%
3066       \ifnumequal{#2}{\tmp}%
3067       {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
3068     }%

```

```

3069     }%
3070     }%
3071     \fi%
3072   }%
3073 }%
3074 %

```

`\printlines` So, we have decided which part of line number sets will be printed depending of these value. Now we are ready to print them. If the lineation is by pstart, we print the pstart. Arguments are 1) start page number 2) start line number 3) start subline number 4) end page number 5) end line number 6) end subline number 7) font specification 8) side flag

```

3075 \def\printlines#1|#2|#3|#4|#5|#6|#7|#8|{%
3076   \begingroup%
3077 %

```

If we use Lua $\TeX$ , ensure we use good text's direction.

```

3078   \ifluatex%
3079     \edef\@tmp{\the\textdir}%
3080     \ifdefstring{\@tmp}{TLT}{\textdir TLT}%Test in order to prevent
    spurious space (bug #397)
3081     \fi%
3082 %

```

Decide which part of line number components we will print.

```

3083   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
3084 %

```

One subtlety left here is when to print a period between numbers. But the only instance in which this is tricky is for the ending sub-line number: it could come after the starting sub-line number (in which case we want only the dash) or after an ending line number (in which case we need to insert a period). So, first, print the start line number.

```

3085   \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
3086     {\bgroup}%
3087     {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
    hfill}%
3088   \ifl@d@pnum%
3089     \wrap@edcrossref{\@this@crossref@start}{#1}%
3090     \csuse{Xsublinesep@\@currentseries}%
3091     \fi%
3092   \wrap@edcrossref{\@this@crossref@start}{%
    \linenumrep{#2}%
3093     \iftoggle{Xlineflag@\@currentseries}{#8}{}%
3094   }%
3095   \ifl@d@ssub%
3096     \csuse{Xsublinesep@\@currentseries}%
3097     \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
3098     \fi
3099   \egroup%
3100 %
3101 %

```

Then print the dash + end line number, or the range symbol.

```

3102 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3103   {\bgroup}%
3104   {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
3105 \ifl@d@Xtwolines%
3106   \ifl@d@Xmorethantwolines%
3107     \csuse{Xmorethantwolines@\@currentseries}%
3108   \else%
3109     \csuse{Xtwolines@\@currentseries}%
3110   \fi%
3111 \else%
3112   \ifl@d@dash%
3113     \ifdefined\linerangesep@%
3114       \linerangesep@%
3115     \else%
3116       \csuse{Xlinerangeseparator@\@currentseries}%
3117     \fi%
3118   \fi%
3119 \ifl@d@pnum%
3120   \wrap@edcrossref{\@this@crossref@end}{#4}%
3121   \csuse{Xsublinesep@\@currentseries}%
3122 \fi%
3123 \ifl@d@elin%
3124   \wrap@edcrossref{\@this@crossref@end}{%
3125     \linenumrep{#5}%
3126     \iftoggle{Xlineflag@\@currentseries}{#8}{}}%
3127   }%
3128 \fi%
3129 \ifl@d@esl%
3130   \ifl@d@elin%
3131     \csuse{Xsublinesep@\@currentseries}%
3132   \fi%
3133   \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}%
3134 \fi%
3135 \fi%
3136 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3137   {}%
3138   {\hfill}%Prevent underfull hbox
3139 \egroup%
3140 \endgroup%
3141 }%
3142 %

```

## XIII Familiar footnotes

### XIII.1 Adjacent footnotes

The original edmac provided users with five series of critical footnotes (`\Afootnote` `\Bfootnote` `\Cfootnote` `\Dfootnote` `\Efootnote`), and  $\LaTeX$  provides a single numbered footnote. The `reledmac` package uses the edmac mechanism to provide six series of numbered footnotes.

First, though, the `footmisc` package has an option whereby two or more consecutive `\footnotes` have their marks separated by commas. This seemed to Peter Wilson such a useful ability that it was provided automatically by `eledmac`.

Maïeul Rouquette has maintained this feature in `reledmac`, despite he thought that is not directly in relationship with the aim of `reledmac`.

`\multiplefootnotemarker` `\multfootsep` These macros may have been defined by the `memoir` class, are provided by the `footmisc` package and perhaps by other footnote packages. That is why we use `\providecommand` and not `\newcommand`.

```
3143 \providecommand*\multiplefootnotemarker}{3sp}
3144 \providecommand*\multfootsep}{\textsuperscript{\normalfont,}}
3145
3146 %
```

`\m@mmf@prepare` A pair of self-cancelling kerns. This may have been defined in the `memoir` class.

```
3147 \providecommand*\m@mmf@prepare}{%
3148 \kern-\multiplefootnotemarker
3149 \kern\multiplefootnotemarker\relax}
3150 %
```

`\m@mmf@check` This may have been defined in the `memoir` class. If it recognises the last kern as `\multiplefootnotemarker` it typesets `\multfootsep`.

```
3151 \providecommand*\m@mmf@check}{%
3152 \ifdim\lastkern=\multiplefootnotemarker\relax
3153 \edef\@x@sf{\the\spacefactor}%
3154 \unkern
3155 \multfootsep
3156 \spacefactor\@x@sf\relax
3157 \fi}
3158
3159 %
```

We have to modify `\@footnotetext` and `\@footnotemark`. However, if `memoir` is used the modifications have already been made.

```
3160 \@ifclassloaded{memoir}{}{%
3161 %
```

`\@footnotetext` Add `\m@mmf@prepare` at the end of `\@footnotetext`.

```
3162 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3163 %
```

`\@footnotemark` Modify `\@footnotemark` to cater for adjacent `\footnotes`.

```
3164
3165 \patchcmd{\@footnotemark}
3166   {\nobreak}
3167   {\m@mmf@check
3168    \nobreak
3169   }
3170   {}{}
3171 \patchcmd{\@footnotemark}
3172   {\@makefnmark}
3173   {\@makefnmark
3174    \m@mmf@prepare
3175   }
3176   {}{}
3177 %
```

Finished the modifications for the non-memoir case.

```
3178 }
3179
3180 %
```

## XIII.2 Regular footnotes for numbered texts

`\l@doldold@footnotetext` In order to enable the regular `\footnotes` in numbered text we have to play around  
`\@footnotetext` with its `\@footnotetext`, using different forms for when in numbered or regular text.

```
3181 \pretocmd{\@footnotetext}{%
3182   \ifnumberedpar@
3183     \edtext{}{\l@dbfnote{#1}}%
3184   \else
3185   }{}{}
3186 \apptocmd{\@footnotetext}{\fi}{}{}%
3187 %
```

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\v1@dbfnote` calls the original  
`\v1@dbfnote` `\@footnotetext`. We also patch `\footnote` in order to get the correct footnote  
`\v1@dbfnote` numbers when typesetting parallel texts. This is moved into a `\get@fnmark` command.  
`\footnote`  
`\get@fnmark`  
`\get@thisfootnote`



```

3188
3189 \patchcmd%
3190   {\footnote}%
3191   {\stepcounter\@mpfn}%
3192   {%
3193   \ifl@dpairing%
3194   \global\advance\footnote@reading by \@ne%
3195     \get@thisfootnote%
3196     \get@fnmark{\thisfootnote}%
3197     \ifcsdef{footnotereading\the\footnote@reading=typeset}%
3198     {\setcounter{\@mpfn}{\csuse{footnotereading\the\footnote@reading=
typeset}}}%
3199     {\setcounter{\@mpfn}{\footnote@reading}}}%
3200   \else%
3201     \stepcounter\@mpfn%
3202   \fi%
3203   }%
3204   {}
3205   {}
3206
3207 \newcommand{\get@thisfootnote}{%
3208   \ifl@dpairing
3209     \protected@xdef\thisfootnote{\the\footnote@reading}%
3210   \else%
3211     \protected@xdef\thisfootnote{\thefootnote}%
3212   \fi%
3213 }%
3214
3215 \newcommand{\l@dbfnote}[1]{%
3216   \get@thisfootnote%
3217   \gdef\@tag{#1\relax}%
3218   \ifledRcol%
3219     \xright@appenditem{%
3220       \ifdefined\Hy@footnote@currentHref%
3221         \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3222       \fi%
3223       \noexpand\vl@dbfnote{\expandonce\@tag}{\thisfootnote}%
3224     }%
3225     \to\inserts@listR
3226     \global\advance\insert@countR \@ne%
3227   \else%
3228     \xright@appenditem{%
3229       \ifdefined\Hy@footnote@currentHref%
3230         \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3231       \fi%
3232       \noexpand\vl@dbfnote{\expandonce\@tag}{\thisfootnote}%
3233     }%
3234     \to\inserts@list

```

```

3235     \global\advance\insert@count \@ne%
3236     \fi
3237     \ignorespaces%
3238 }%
3239
3240 \newcommand{\get@fnmark}[1]{%
3241     \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
3242         l@dprintingcolumns}}%
3243         {%
3244             \stepcounter{footnote@typeset}%
3245             \setcounter{footnote}{\c@footnote@typeset}%
3246             \immediate\write\@mainaux{%
3247                 \global\csdef{footnotereading#1=typeset}{\the\c@footnote@typeset}
3248             }%
3249             \def\@thefnmark{\thefootnote}%
3250             {%
3251                 \@namedef{@thefnmark}{#1}%
3252             }%
3253 }%
3254
3255 \newcommand{\v1@dbfnote}[2]{%
3256     \get@fnmark{#2}%
3257     \@footnotetext{#1}%
3258 }%
3259 %

```

### XIII.3 Footnote formats

Some of the code for the various formats is remarkably similar to that in section ??.

The following macros generally set things up for the ‘standard’ footnote format.

`\prebodyfootmark` Two convenience macros for use by `\...@footnotemark...` macros.  
`\postbodyfootmark`

```

3260 \newcommand*\prebodyfootmark{%
3261     \leavevmode
3262     \ifhmode
3263         \edef\@x@sf{\the\spacefactor}%
3264         \m@mmf@check
3265         \nobreak
3266     \fi}
3267 \newcommand*\postbodyfootmark{%
3268     \m@mmf@prepare
3269     \ifhmode\spacefactor\@x@sf\fi\relax}
3270
3271 %

```

## XIII.4 Footnote arrangement

### XIII.4.1 User level macro

`\arrangementX` `\arrangementX[⟨s⟩]{⟨arrangement⟩}` command calls, for each series, a specific command which set many counters and commands in order to define specific arrangement.

```

3272 \newcommandx{\arrangementX}[2][1,usedefault]{%
3273   \def\do##1{%
3274     \csname arrangementX@#2\endcsname{##1}%
3275   }%
3276   \ifstrempy{#1}%
3277     {%
3278     \dolistloop{\@series}%
3279     }%
3280     {
3281     \docsvlist{#1}%
3282     }%
3283 }%
3284 %

```

### XIII.4.2 Normal footnotes

`\normal@footnotemarkX` `\normal@footnotemarkX{⟨series⟩}` sets up the typesetting of the marker at the point where the footnote is called for.

```

3285 \newcommand*{\normal@footnotemarkX}[1]{%
3286   \prebodyfootmark
3287   \wrapped@bodyfootmarkX{#1}%
3288   \postbodyfootmark}
3289
3290 %

```

`\normalbodyfootmarkX` The `\normalbodyfootmarkX{⟨series⟩}` *really* typesets the in-text marker. The style is the normal superscript.

```

3291 \newcommand*{\normalbodyfootmarkX}[1]{%
3292   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3293 %

```

`\normalvfootnoteX` `\normalvfootnoteX{⟨series⟩}{⟨text⟩}` does the `\insert` for the `⟨series⟩` and calls the series' `\footfmt...` to format the `⟨text⟩`.

```

3294 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnoteX}[2]{%
3295   \insert\@nameuse{footins#1}\bgroup
3296   \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
3297   \noindent\csuse{hooknoteX@#1}%
3298   \csuse{notefontsizeX@#1}%
3299   \footsplitskips
3300   \ifl@dpairing\ifl@dpageing\else%
3301     \setnoteswidthliketwocolumnsX@{#1}%

```

```

3302 \fi\fi%
3303 \setnotesXpositionliketwocolumns@{#1}%
3304 \spaceskip=\z@skip \xspaceskip=\z@skip
3305 \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3306
3307 %

```

`\mpnormalvfootnoteX` The minipage version.

```

3308 \newcommand*{\mpnormalvfootnoteX}[2]{%
3309 \get@thisfootnoteX{#1}%
3310 \get@fnmarkX{#1}{\thisfootnote}%
3311 \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3312 \global\setbox\@nameuse{mpfootins#1}\vbox{%
3313 \unvbox\@nameuse{mpfootins#1}
3314 \noindent\csuse{bhooknoteX@#1}%
3315 \csuse{notefontsizeX@#1}%
3316 \hsize\columnwidth
3317 \@parboxrestore
3318 \color@begingroup
3319 \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3320
3321 %

```

`\normalfootfmtX` `\normalfootfmtX{<series>}{<text>}` typesets the footnote text, prepended by the marker.

```

3322 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
3323 \ifluatex%
3324 \textdir\footnote@luatextextdir%
3325 \pardir\footnote@luatexpardir%
3326 \par%
3327 \fi%
3328 \protected@edef\@currentlabel{%
3329 \@nameuse{@thefnmark#1}%
3330 }%
3331 \ledsetnormalparstuffX{#1}%
3332 \hangindent=\csuse{hangindentX@#1}%
3333 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3334 {\csuse{notenunfontX@#1}\wrapped@footfootmarkX{#1}}\strut%
3335 #2\strut\par}}
3336
3337 %

```

`\normalfootfootmarkX` `\normalfootfootmarkX{<series>}` is called by `\normalfootfmtX` to typeset the footnote marker in the footer before the footnote text.

```

3338 \newcommand*{\normalfootfootmarkX}[1]{%
3339 \textsuperscript{\@nameuse{@thefnmark#1}}}
3340
3341 %

```

`\normalfootstartX` `\normalfootstartX{<series>}` is the `<series>` footnote starting macro used in the output routine.

```

3342 \newcommand*{\normalfootstartX}[1]{%
3343   \ifdimequal{Opt}{\prenotesX@}{}%
3344   {%
3345     \iftoggle{prenotesX@}{%
3346       \togglefalse{prenotesX@}%
3347       \skip\csname footins#1\endcsname=%
3348       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3349     }%
3350   }%
3351 }%
3352 \vskip\skip\csname footins#1\endcsname%
3353 \leftskip=\z@
3354 \rightskip=\z@
3355 \ifl@dpairing\else%
3356   \hsize=\old@hsize%
3357 \fi%
3358 \setnoteswidthliketwocolumnsX@{#1}%
3359 \setnotesXpositionliketwocolumns@{#1}%
3360 \print@footnoteXrule{#1}%
3361 }%
3362
3363 %

```

`\normalfootnoteruleX` The rule drawn before the footnote series group.

```

3364 \let\normalfootnoteruleX=\footnoterule
3365
3366 %

```

`\normalfootgroupX` `\normalfootgroupX{<series>}` sends the contents of the `<series>` insert box to the output page without alteration.

```

3367 \newcommand*{\normalfootgroupX}[1]{%
3368   \csuse{bhookgroupX@#1}%
3369   \unvbox\@nameuse{footins#1}%
3370   \hsize=\old@hsize%
3371 }%
3372
3373 %

```

`\mpnormalfootgroupX` The minipage version.

```

3374 \newcommand*{\mpnormalfootgroupX}[1]{%
3375   \vskip\skip\@nameuse{mpfootins#1}
3376   \ifl@dpairing\ifparledgroup%
3377     \leavevmode\marks\parledgroup@{begin}%
3378     \marks\parledgroup@series{#1}%

```

```

3379 \marks\parledgroup@type{footnoteX}%
3380 \fi\fi\normalcolor
3381 \ifparledgroup%
3382 \ifl@dpairing%
3383 \else%
3384 \setnoteswidthliketwocolumnsX@{#1}%
3385 \setnotesXpositionliketwocolumns@{#1}%
3386 \print@footnoteXrule{#1}%
3387 \fi%
3388 \else%
3389 \setnoteswidthliketwocolumnsX@{#1}%
3390 \setnotesXpositionliketwocolumns@{#1}%
3391 \print@footnoteXrule{#1}%
3392 \fi%
3393 \csuse{bhookgroupX@#1}%
3394 \unvbox\@nameuse{mpfootins#1}}
3395
3396 %

```

### `\normalbfnoteX`<sup>97</sup>

```

3398 \newcommand{\normalbfnoteX}[2]{%
3399 \get@thisfootnoteX{#1}%
3400 \ifledRcol%
3401 \ifluatex
3402 \footnotelang@lua[R]%
3403 \fi
3404 \@ifundefined{xpg@main@language}%if polyglossia
3405 {}%
3406 {\footnotelang@poly[R]}%
3407 \xright@appenditem{%
3408 \noexpand\led@set@index@fornote{#1}%
3409 \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%
3410 \noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}%
3411 \noexpand\led@reinit@index@fornote%
3412 }%
3413 \to\inserts@listR
3414 \global\advance\insert@countR \@ne%
3415 \else%
3416 \ifluatex
3417 \footnotelang@lua%
3418 \fi
3419 \@ifundefined{xpg@main@language}%if polyglossia
3420 {}%
3421 {\footnotelang@poly}%
3422 \xright@appenditem{%
3423 \noexpand\led@set@index@fornote{#1}%
3424 \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%

```

```

3425 \noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}%
3426 \noexpand\led@reinit@index@fornote%
3427 }%
3428 \to\inserts@list
3429 \global\advance\insert@count \@ne%
3430 \fi
3431 \ignorespaces}
3432 %
3433 %

```

**\get@thisfootnoteX** The macro `\get@thisfootnote` command just saves the footnote number in the `\thisfootnote` macro, depending on the use of pairing environments.

```

3434 \newcommand{\get@thisfootnoteX}[1]{%
3435 \ifl@dpairing%
3436 \protected@xdef\thisfootnote{\the\csname footnote#1@reading\endcsname
3437 }%
3438 \else%
3439 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3440 \fi%
3441 }%

```

**\vbfnoteX** This command calls the correct footnote-inserting commands.

```

3442 \newcommand{\vbfnoteX}[3]{%
3443 \get@fnmarkX{#1}{#3}%
3444 \@nameuse{regvfootnote#1}{#1}{#2}%
3445 }%
3446 %
3447 %

```

**\get@fnmarkX** This command gets the correct footnote number when typesetting parallel texts.

```

3448 \newcommand{\get@fnmarkX}[2]{%
3449 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
3450 l@dprintingcolumns}}%
3451 {%
3452 \stepcounter{footnote#1@typeset}%
3453 \setcounter{footnote#1}{\value{footnote#1@typeset}}%
3454 \@namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
3455 \immediate\write\@mainaux{%
3456 \global\csdef{footnote#1reading#2=typeset}{\the\csname c@footnote
3457 #1@typeset\endcsname}%
3458 }%
3459 }%
3460 {%
3461 \@namedef{@thefnmark#1}{#2}%
3462 }%

```

```

3462 %
3463 %

\vnumfootnoteX_{2}{%
3465   \ifnumberedpar@
3466   \edtext{}{\normalbfnoteX{#1}{#2}}%
3467   \else
3468   \def\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3469   \@nameuse{regvfootnote#1}{#1}{#2}%
3470   \fi}
3471
3472 %

```

`arrangementX@normal` `\arrangementX@normal{<series>}` initialises the settings for the `<series>` footnotes. This should always be called for each series.

```

3473 \newcommand*{\arrangementX@normal}[1]{%
3474   \csgdef{series@displayX#1}{normal}
3475   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3476   \expandafter\newcount\csname prevpage#1@num\endcsname
3477   \@namedef{footnotemark#1}{\normal@footnotemarkX{#1}}
3478   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3479   \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3480   \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
3481   \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3482   \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3483   \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3484   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3485   \count\csname footins#1\endcsname=1000
3486   \csxdef{default@footins#1}{1000}%Use to have note only for one side
3487   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3488   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3489   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3490 %

```

Additions for minipages.

```

3491   \ifnoledgroup@ \else%
3492     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3493     \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3494     \count\csname mpfootins#1\endcsname=1000
3495     \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3496     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3497     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3498   \fi
3499 }
3500
3501 %

```



## XIII.4.3 Two columns footnotes

The following macros set footnotes in two columns. It is assumed that the length of each footnote is less than the column width.

```

\arrangementX@twocol  \newcommand*\arrangementX@twocol}[1]{%
3503   \csgdef{series@displayX#1}{twocol}
3504   \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
3505   \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3506   \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3507   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3508   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3509   \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3510   \twocolfootsetupX{#1}
3511   \ifnoledgroup@ \else%
3512     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3513     \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
3514     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3515     \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3516     \mptwocolfootsetupX{#1}
3517   \fi%
3518 }
3519
3520 %

\twocolfootsetupX  \twocolfootsetupX{<series>}
\mptwocolfootsetupX
3521 \newcommand*\twocolfootsetupX}[1]{%
3522   \count\csname footins#1\endcsname 500
3523   \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3524   \multiply\dimen\csname footins#1\endcsname by \tw@}
3525 \newcommand*\mptwocolfootsetupX}[1]{%
3526   \count\csname mpfootins#1\endcsname 500
3527   \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
3528
3529 %

\twocolvfootnoteX  \twocolvfootnoteX{<series>}
3530 \notbool{parapparatus@}{\newcommand*}{\newcommand*}{\twocolvfootnoteX}[2]{%
3531   \insert\csname footins#1\endcsname\bgroup%
3532     \hspace=\expandafter\dimexpr\csuse{widthX@#1}\relax%
3533     \noindent\csuse{bhooknoteX@#1}%
3534     \csuse{notefontsizeX@#1}%
3535     \footsplitskips%
3536     \spaceskip=\z@skip \xspaceskip=\z@skip%
3537     \@nameuse{footfmt#1}{#1}{#2}\egroup}
3538
3539 %

```

```

\twocolfootfmtX \twocolfootfmtX{<series>}
3540 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3541   \protected@edef\@currentlabel{%
3542     \@nameuse{@thefnmark#1}%
3543   }%
3544   \normal@pars%
3545   \hangindent=\csuse{hangindentX@#1}%
3546   \everypar{\hangindent=\csuse{hangindentX@#1}}%
3547   \hsize \csuse{hsizetwocolX@#1}%
3548   \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3549   \tolerance=5000\relax%
3550   \par%
3551   \@tempdima=\parindent%
3552   \csuse{colalignX@#1}%
3553   \parindent=\@tempdima%
3554   {\hspace{\parindent}%
3555    \csuse{notenumpfontX@#1}\wrapped@footfootmarkX{#1}\strut%
3556    #2\strut\par}%
3557   \allowbreak%
3558 }%
3559 %
3560 %

```

```

\twocolfootgroupX \twocolfootgroupX{<series>}
\mptwocolfootgroupX
3561 \newcommand*{\twocolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3562   \splittopskip=\ht\strutbox
3563   \expandafter
3564   \rigidbalanceX\csname footins#1\endcsname \tw@ \splittopskip}}
3565
3566 \newcommand*{\mptwocolfootgroupX}[1]{%
3567   \vskip\skip\@nameuse{mpfootins#1}
3568   \ifl@dpairing\ifparledgroup%
3569     \leavevmode\marks\parledgroup@{begin}%
3570     \marks\parledgroup@series{#1}%
3571     \marks\parledgroup@type{footnoteX}%
3572     \fi\fi\normalcolor
3573     \ifparledgroup%
3574       \ifl@dpairing%
3575       \else%
3576         \setnoteswidthliketwocolumnsX@{#1}%
3577         \setnotesXpositionliketwocolumns@{#1}%
3578         \print@footnoteXrule{#1}%
3579       \fi%
3580     \else%
3581       \setnoteswidthliketwocolumnsX@{#1}%
3582       \setnotesXpositionliketwocolumns@{#1}%
3583       \print@footnoteXrule{#1}%

```

```

3584 \fi%
3585 \csuse{bhookgroupX@#1}%
3586 \splittopskip=\ht\strutbox
3587 \expandafter
3588 \rigidbalanceX\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3589
3590 %

```

#### XIII.4.4 Three columns footnotes

The following macros set footnotes in three columns. It is assumed that the length of each footnote is less than the column width.

```

\arrangementX@threecol} \newcommand*{\arrangementX@threecol}[1]{%
3592 \csgdef{series@displayX#1}{threecol}
3593 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3594 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3595 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3596 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3597 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3598 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3599 \threecolfootsetupX{#1}
3600 \ifnoledgroup@ \else%
3601 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3602 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3603 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3604 \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3605 \mpthreecolfootsetupX{#1}
3606 \fi%
3607 }
3608
3609 %

```

```
\threecolfootsetupX \threecolfootsetupX{<series>}
```

```
\mpthreecolfootsetupX
```

```

3610 \newcommand*{\threecolfootsetupX}[1]{%
3611 \count\csname footins#1\endcsname 333
3612 \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3613 \multiply\dimen\csname footins#1\endcsname by \thr@@}
3614 \newcommand*{\mpthreecolfootsetupX}[1]{%
3615 \count\csname mpfootins#1\endcsname 333
3616 \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}
3617
3618 %

```

```
\threecolvfootnoteX \threecolvfootnoteX{<series>}{<text>}
```

```

3619 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootnoteX}[2]{
%
3620   \insert\csname footins#1\endcsname\bgroup%
3621   \hsize=\expandafter\dimexpr\csuse{widthX@#1}\relax%
3622   \noindent\csuse{bhooknoteX@#1}%
3623   \csuse{notefontsizeX@#1}%
3624   \footsplitskips%
3625   \@nameuse{footfmt#1}{#1}{#2}\egroup}
3626
3627 %

```

`\threecolfootfmtX` `\threecolfootfmtX{(series)}`

```

3628 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3629   \protected@edef\@currentlabel{%
3630     \@nameuse{@thefnmark#1}%
3631   }%
3632   \hangindent=\csuse{hangindentX@#1}%
3633   \everypar{\hangindent=\csuse{hangindentX@#1}}%
3634   \normal@pars%
3635   \hsize \csuse{hsizethreecolX@#1}%
3636   \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3637   \tolerance=5000\relax%
3638   \@tempdima=\parindent%
3639   \csuse{colalignX@#1}%
3640   \parindent=\@tempdima%
3641   {\hspace{\parindent}}%
3642   \csuse{notenunfontX@#1}\wrapped@footfootmarkX{#1}\strut%
3643   #2\strut\par}\allowbreak}
3644
3645 %

```

`\threecolfootgroupX` `\threecolfootgroupX{(series)}`  
`\mpthreecolfootgroupX`

```

3646 \newcommand*{\threecolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3647   \splittopskip=\ht\strutbox
3648   \expandafter
3649   \rigidbalanceX\csname footins#1\endcsname \thr@@ \splittopskip}}
3650
3651 \newcommand*{\mpthreecolfootgroupX}[1]{%
3652   \vskip\skip\@nameuse{mpfootins#1}
3653   \ifl@dpairing\ifparledgroup
3654     \leavevmode\marks\parledgroup@{begin}%
3655     \marks\parledgroup@series{#1}%
3656     \marks\parledgroup@type{footnoteX}%
3657   \fi\fi\normalcolor
3658   \ifparledgroup%
3659     \ifl@dpairing%
3660     \else%

```

```

3661     \setnoteswidthliketwocolumnsX@{#1}%
3662     \setnotesXpositionliketwocolumns@{#1}%
3663     \print@footnotexrule{#1}%
3664     \fi%
3665 \else%
3666     \setnoteswidthliketwocolumnsX@{#1}%
3667     \setnotesXpositionliketwocolumns@{#1}%
3668     \print@footnotexrule{#1}%
3669     \fi%
3670     \csuse{bhookgroupX@#1}%
3671     \splittopskip=\ht\strutbox
3672     \expandafter
3673     \rigidbalanceX\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3674
3675 %

```

### XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

`\arrangementX@threecol` `\footparagraphX{<series>}`

```

3676 \newcommand*{\arrangementX@paragraph}[1]{%
3677   \csgdef{series@displayX#1}{paragraph}%
3678   \expandafter\newcount\csname #1prevpage@num\endcsname
3679   \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3680   \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
3681   \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3682   \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3683   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3684   \count\csname footins#1\endcsname=1000
3685   \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3686   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3687   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3688   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3689   \para@footsetupX{#1}
3690   \ifnoledgroup@else
3691     \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
3692     \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
3693     \count\csname mpfootins#1\endcsname=1000
3694     \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3695     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3696     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3697   \fi
3698 }
3699
3700 %

```

`\para@footsetupX` `\para@footsetupX{<series>}`

```

3701 \newcommand*\para@footsetupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3702 \setnoteswidthliketwocolumnsX@{#1}%
3703 \ifcsemtyp{widthX@#1}%
3704   {}%
3705   {\columnwidth=\expandafter\dimexpr\csuse{widthX@#1}\relax}%
3706 \dimen0=\baselineskip
3707 \multiply\dimen0 by 1024
3708 \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
%
3709 \expandafter
3710 \xdef\csname footfudgefactor#1\endcsname{%
3711   \expandafter\strip@pt\dimen0 }}
3712
3713 %

```

`\parafootstartX` `\parafootstartX{<series>}`

```

3714 \newcommand*\parafootstartX}[1]{%
3715   \ifdimequal{Opt}{\prenotesX@}{}%
3716   {%
3717     \iftoggle{prenotesX@}{%
3718       \togglefalse{prenotesX@}%
3719       \skip\csname footins#1\endcsname=%
3720       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3721     }%
3722   }%
3723 }%
3724 \leftskip=\z@
3725 \rightskip=\z@
3726 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3727 \vskip\skip\@nameuse{footins#1}%
3728 \setnoteswidthliketwocolumnsX@{#1}%
3729 \setnotesXpositionliketwocolumns@{#1}%
3730 \print@footnoteXrule{#1}%
3731 }
3732
3733 %

```

`\para@vfootnoteX` `\para@vfootnoteX{<series>}{<text>}`  
`\mppara@vfootnoteX`

```

3734 \newcommand*\para@vfootnoteX}[2]{%
3735 \insert\csname footins#1\endcsname%
3736 \bgroup
3737 \csuse{notefontsizeX@#1}
3738 \footsplitskips
3739 \setbox0=\vbox{\hsize=\maxdimen%
3740 \let\bidir@RTL@everypar\@empty%
3741 \noindent\csuse{bhooknoteX@#1}%
3742 \@nameuse{footfmt#1}{#1}{#2}}%

```

```

3743 \setbox0=\hbox{\unvxhX{0}{#1}}%
3744 \dp0=\z@
3745 \ht0=\csname footfudgefactor#1\endcsname\wd0
3746 \box0
3747 \penalty0
3748 \egroup}
3749 \newcommand*\mppara@vfootnoteX}[2]{%
3750 \get@thisfootnoteX{#1}%
3751 \get@fnmarkX{#1}{\thisfootnote}%
3752 \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3753 \global\setbox\@nameuse{mpfootins#1}\vbox{%
3754 \unvbox\@nameuse{mpfootins#1}
3755 \csuse{notefontsizeX@#1}
3756 \footsplitskips
3757 \setbox0=\vbox{\hsize=\maxdimen%
3758 \let\ bidi@RTL@everypar\@empty%
3759 \noindent\color@begingroup%
3760 \csuse{bhooknoteX@#1}%
3761 \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3762 \setbox0=\hbox{\unvxhX{0}{#1}}%
3763 \dp0=\z@
3764 \ht0=\csname footfudgefactor#1\endcsname\wd0
3765 \box0
3766 \penalty0}}
3767
3768 %

```

```

\unvxhX69 \newcommand*\unvxhX}[2]{% 2th is optional for retro-compatibility
3770 \setbox0=\vbox{\unvbox#1%
3771 \global\setbox1=\lastbox}%
3772 \unhbox1
3773 \unskip % remove \rightskip,
3774 \unskip % remove \parfillskip,
3775 \unpenalty % remove \penalty of 10000,
3776 \hskip\csuse{afternoteX@#2}} % but add the glue to go between the notes
3777
3778 %

```

`\parafootfmtX` `\parafootfmtX{<series>}`

```

3779 \newcommand*\parafootfmtX}[2]{%
3780 \protected@edef\@currentlabel{%
3781 \@nameuse{@thefnmark#1}%
3782 }%
3783 \insertparafootsepX{#1}%
3784 \ledsetnormalparstuff@common%
3785 {\csuse{notenumfontX@#1}%
3786 \csuse{notenumfontX@#1}%
3787 \wrapped@footfootmarkX{#1}%

```

```

3788 \strut%
3789 #2\penalty-10}}
3790
3791 %

```

`\para@footgroupX` `\para@footgroupX{<series>}`  
`\mppara@footgroupX`

```

3792 \newcommand*\para@footgroupX}[1]{%
3793 \hspace=\expandafter\dimexpr\csuse{widthX@#1}\relax%
3794 \unvbox\csname footins#1\endcsname
3795 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3796 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3797 \makeboxofhboxes
3798 \setbox0=\hbox{\unhbox0 \removehboxes}%
3799 \csuse{hookgroupX@#1}
3800 \csuse{notefontsizeX@#1}
3801 \unhbox0\par}
3802
3803 \newcommand*\mppara@footgroupX}[1]{%
3804 \setnoteswidthliketwocolumnsX@{#1}%
3805 \vskip\skip\@nameuse{mpfootins#1}
3806 \ifl@dpairing\ifparledgroup
3807 \leavevmode%
3808 \leavevmode\marks\parledgroup@{begin}%
3809 \marks\parledgroup@series{#1}%
3810 \marks\parledgroup@type{footnoteX}%
3811 \fi\fi\normalcolor
3812 \ifparledgroup%
3813 \ifl@dpairing%
3814 \else%
3815 \setnoteswidthliketwocolumnsX@{#1}%
3816 \setnotesXpositionliketwocolumns@{#1}%
3817 \print@footnoteXrule{#1}%
3818 \fi%
3819 \else%
3820 \setnoteswidthliketwocolumnsX@{#1}%
3821 \setnotesXpositionliketwocolumns@{#1}%
3822 \print@footnoteXrule{#1}%
3823 \fi%
3824 \unvbox\csname mpfootins#1\endcsname
3825 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3826 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3827 \makeboxofhboxes
3828 \setbox0=\hbox{\unhbox0 \removehboxes}%
3829 \csuse{hookgroupX@#1}%
3830 \csuse{notefontsizeX@#1}%
3831 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3832 \unhbox0\par}}
3833
3834 %

```



**Insertion of the footnotes separator** The command `\insertparafootsepX{<series>}` must be called at the beginning of `\parafootftmX`.

```

\prevpage@num  \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep  \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3837             {\csuse{parafootsepX@#1}}%
3838             {}%
3839         }
3840     %

```

### XIII.5 Wrapping footnote marks in hyperlink

`\wrapped@footfootmarkX` `\wrapped@footfootmarkX` prints the footnote mark of the footpage, wrapped in hyperref package's commands, if needed.

```

3841 \newcommand{\wrapped@footfootmarkX}[1]{%
3842     \ifdefined\hypertarget%
3843         \hyperlink%
3844             {@bodyfootmark#1@this@footnoteX@reading}%
3845             {@\nameuse{footfootmark#1}}%
3846         \Hy@raisedlink{%
3847             \hypertarget%
3848                 {@footnotemark#1@this@footnoteX@reading}%
3849                 {}%
3850             }%
3851         \else%
3852             \@nameuse{footfootmark#1}%
3853         \fi%
3854     }%
3855     %

```

`\wrapped@bodyfootmarkX` `\wrapped@bodyfootmarkX` prints the footnote mark of the text body, wrapped in hyperref package's commands, if needed.

```

3856 \newcommand{\wrapped@bodyfootmarkX}[1]{%
3857     \ifdefined\hypertarget%
3858         \hyperlink%
3859             {@footnotemark#1\expandafter\the\csname footnote#1@reading\
endcsname}%
3860             {@\nameuse{bodyfootmark#1}}%
3861         \Hy@raisedlink{%
3862             \hypertarget%
3863                 {@bodyfootmark#1\expandafter\the\csname footnote#1@reading\
endcsname}%
3864                 {}%
3865             }%
3866         \else%
3867             \@nameuse{bodyfootmark#1}%
3868         \fi%

```

```
3869 }%
3870 %
```

## XIV Code common to both critical and familiar footnote in normal arrangement

`\par` should always be redefined to `\endgraf` within the format macro (this is what `\normal@pars` does), to override tricky material in the main text to get the lines numbered automatically (as set up by `\autopar`, for example).

In the case of footnote arranged in a “normal” way, we also must set some setting for paragraph indent and text direction when using Lua $\TeX$ .

That why we have defined `\ledsetnormalparstuff@common` in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

```
\ledsetnormalparstuff@common  \newcommand*\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff        \ifluatex%
\ledsetnormalparstuffX       \textdir\footnote@luatextextdir%
3874 \pardir\footnote@luatexpardir%
3875 \fi%
3876 \csuse{\csuse{footnote@dir}}%
3877 \normal@pars%
3878 \parfillskip \z@ \@plus 1fil}%
3879
3880 \newcommand*\Xledsetnormalparstuff}[1]{%
3881 \ledsetnormalparstuff@common%
3882 \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%
3883 }%
3884
3885 \newcommand*\ledsetnormalparstuffX}[1]{%
3886 \ledsetnormalparstuff@common%
3887 \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%
3888 }%
3889 %
```

## XV Footnotes' width for two columns

We define here some commands which make sense only with `reledpar`, but must be called when defining notes parameters. These commands change the width of block notes to allow them to have the same size than two parallel columns.

```
\old@hsize  These two commands are called at the beginning of critical or familiar notes groups.
\setXnoteswidthliketwocolumns@ They set, if the option is enabled, the \hspace. They are also called at the on the setup
\setnoteswidthliketwocolumnsX@ for paragraphed notes.
```

```

3890
3891 \newdimen\old@hsize%
3892 \AtBeginDocument{\old@hsize=\hsize}%
3893
3894 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3895   \global\let\hsize@fornote=\hsize%
3896   \global\old@hsize=\hsize%
3897   \let\old@columnwidth=\columnwidth%
3898   \iftoggle{Xnoteswidthliketwocolumns@#1}%
3899     {%
3900     \csuse{setwidthliketwocolumns@\columns@position}%
3901     \global\let\hsize@fornote=\hsize%
3902     }%
3903     {}%
3904   \let\hsize=\hsize@fornote%
3905   \let\columnwidth=\old@columnwidth%
3906 }%
3907
3908 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3909   \global\let\hsize@fornote=\hsize%
3910   \global\old@hsize=\hsize%
3911   \let\old@columnwidth=\columnwidth%
3912   \iftoggle{noteswidthliketwocolumnsX@#1}%
3913     {%
3914     \csuse{setwidthliketwocolumns@\columns@position}%
3915     \global\let\hsize@fornote=\hsize%
3916     }%
3917     {}%
3918   \let\hsize=\hsize@fornote%
3919   \let\columnwidth=\old@columnwidth%
3920 }%
3921
3922 %

```

`\setnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sense without `reledpar`.

```

3923 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3924   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3925     \csuse{setnotespositionliketwocolumns@\columns@position}%
3926   }{}%
3927 }%
3928
3929 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3930   \iftoggle{noteswidthliketwocolumnsX@#1}{%
3931     \csuse{setnotespositionliketwocolumns@\columns@position}%
3932   }{}%
3933 }%

```

```
3934
3935 %
```

## XVI Footnotes' order

`\fnpos` and `\mpfnpos` simply place their arguments in `\@fnpos` and `\@mpfnpos`, which will be used later in the output routine.

```
\fnpos
\mpfnpos
\@fnpos
\@mpfnpos
3936 \def\@fnpos{familiar-critical}
3937 \def\@mpfnpos{critical-familiar}
3938 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3939 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3940 %
```

## XVII Footnotes' rule

Because the footnotes' rules can be shifted to the right when footnotes are set like two columns, we do not print them directly, but we put them in a `\vbox`.

```
\print@Xfootnoterule41 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnoteXrule42 \vskip-\csuse{Xafterrule@#1}%Because count in \dimen\csuse{#1footins}
3943 \nointerlineskip%
3944 \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
3945 \nointerlineskip%
3946 \vskip\csuse{Xafterrule@#1}%
3947 }%
3948
3949 \newcommand{\print@footnoteXrule}[1]{%
3950 \vskip-\csuse{afterruleX@#1}%Because count in \dimen\csuse{footins#1}
3951 \nointerlineskip%
3952 \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
3953 \nointerlineskip%
3954 \vskip\csuse{afterruleX@#1}%
3955 }%
3956
3957 %
```

## XVIII Specific skip for first series of footnotes

### XVIII.0.1 Overview

`\Xbeforenotes` inserts a specific skip for the first series of notes in a page. As we can't know in advance which series will be the first, we call `\prepare@preXnotes` before inserting any critical notes, in order to prevent page number overlapping.

1. If it is the first note of the current page, it changes the footnote skip for the series to the value specified to `\Xbeforenotes`. It also keeps the series of the note as the first one of the current page.
2. If it is not the first note of the current page:
  - If the current series is printed after the series kept as the first of the current page, then nothing happens.
  - If the current series is printed before the series kept as the first of the current page, then it changes the footnote skip of the current series to the value normally used by the series which was marked as the first of the page. It also keeps the current series as the new first one of the current page.

For example, suppose the series order is A,B. We call first a `\Bfootnote` and a `\Afootnote`. The only skips used are, finally, the skip specific to the first series of the page, and the skip for the B series. If we have not called `\Afootnote`, the only skip used is the skip specific to the first series of the page.

That is perfect.

The series skip and the first series of the current page are reset before the footnotes are printed. Then, the `footstart` macros manage the problem of the first series of the page.

After the rule, the space which is defined by `\Xafterrule` does not depend on whether the series is the first one of the page or not. So we use its normal value for each series.

And now, implementation !

### XVIII.0.2 User level command

`\preXnotes@` If user redefines `\preXnotes@`, via `\preXnotes` to a value greater than 0 pt, this skip will be added before first series notes instead of the notes skip.

```

3958 \newtoggle{preXnotes@}
3959 \toggletrue{preXnotes@}
3960 \newcommand{\preXnotes@}{0pt}
3961 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3962 %

```

The same, but for familiar footnotes.

```

\preXnotes@_63 \newtoggle{prenotesX@}
\preXnotes@_64 \toggletrue{prenotesX@}
3965 \newcommand{\prenotesX@}{0pt}
3966 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
3967 %

```

## XVIII.0.3 Internal commands

```

firstXseries@ \gdef\firstXseries@{}
prepare@preXnotes \newcommand{\prepare@preXnotes}[1]{%
3970   \ifdimequal{0pt}{\preXnotes@}%
3971   {}%
3972   {%
3973     \IfStrEq{\firstXseries@}{-}{%
3974       \global\skip\csuse{#1footins}=\preXnotes@%
3975       \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3976       \gdef\firstXseries@{#1}%
3977     }%
3978     {%
3979       \ifseriesbefore{#1}{\firstXseries@}%
3980       {%
3981         \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3982         \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3983         \gdef\firstXseries@{#1}%
3984       }%
3985     }%
3986   }%
3987 }%
3988 }
3989 %

```

The same thing is required for familiar notes and \prenotesX.

```

firstseriesX@ \gdef\firstseriesX@{}
prepare@prenotesX \newcommand{\prepare@prenotesX}[1]{%
3992   \ifdimequal{0pt}{\prenotesX@}%
3993   {}%
3994   {%
3995     \IfStrEq{\firstseriesX@}{-}{%
3996       \global\skip\csuse{footins#1}=\prenotesX@%
3997       \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3998       \gdef\firstseriesX@{#1}%
3999     }%
4000     {%
4001       \ifseriesbefore{#1}{\firstseriesX@}%
4002       {%
4003         \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
4004         \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
4005         \gdef\firstXseries@{#1}%
4006       }%
4007     }%

```

```

4008 }%
4009 }%
4010 }
4011 %

```

## XIX Endnotes

First, check the noend option.

```

4012 \ifbool{noend@}{}{%Used instead of \ifnoend@ to prevent expansion problem
4013 %

```

`\l@dend@open` `\l@dend@open` and `\l@dend@close` are the macros that are used to open and close the endnote file. Note that all our writing to this file is `\immediate`: all page and line numbers for the endnotes are generated by the same mechanism we use for the footnotes, so that there is no need to defer any writing to catch information from the output routine. The argument of these two command is the series letter.

```

4014 \newcommand{\l@dend@open}[1]{%
4015   \global\booltrue{l@dend@#1}%
4016   \expandafter\immediate%
4017   \expandafter\openout%
4018   \csname l@d@#1end\endcsname%
4019   =\jobname.#1end\relax%
4020 }%
4021 \newcommand{\l@dend@close}[1]{%
4022   \global\boolfalse{l@dend@#1}%
4023   \expandafter\immediate%
4024   \expandafter\closeout\csname l@d@#1end\endcsname%
4025 }%
4026
4027 %

```

`\l@dend@stuff` `\l@dend@stuff` is used by `\beginnumbering` to do everything that is necessary for the endnotes at the start of each section: it opens the `\l@d@end` file, if necessary, and writes the section number to the endnote file.

```

4028 \newcommand{\l@dend@stuff}{%
4029   \def\do##1{%
4030     \ifbool{l@dend@##1}{%
4031       {\l@dend@open{##1}}%
4032       \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{the\section@num}}%
4033     }%
4034     \dolistloop{@series}%
4035   }%
4036
4037 %

```

`\endprint` The `\endprint` here is nearly identical in its functioning to `\normalfootfmt`.  
`\l@d@section` The endnote file also contains `\l@d@section` commands, which supply the section numbers from the main text; standard `reledmac` does nothing with this information, but it is there if you want to write custom macros to do something with it. Arguments are:

- #1 Line numbers and font selection.
- #2 Lemma.
- #3 Note content.
- #4 Series.
- #5 Optional argument of `\Xendnote`.
- #6 Side (L or R).
- #7 Label for cross-referencing.

```

4038 \global\newbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5#6#7{#{%
4039   \csuse{Xendhooknote@#4}%
4040   \csuse{Xendnotefontsize@#4}%
4041   \hangindent=\csuse{Xendhangindent@#4}%
4042   \ifXendinsertsep%
4043     \hskip\csuse{Xendafternote@#4}%
4044     \csuse{Xendsep@#4}%
4045   \else%
4046     \iftoggle{Xendparagraph@#4}%
4047       {\global\Xendinsertsep@true}%
4048       {}%
4049   \fi%
4050   \xdef\@currentseries{#4}%
4051   \def\do##1{%
4052     \setkeys[mac]{truefootnoteoption}{##1}%
4053   }%
4054   \notblank{#5}{\docsvlist{#5}}{ }%
4055   \IfStrEq{#6}{R}{\ledRcol@true}{ }%
4056   \def\@this@crossref@start{#7:start}%
4057   \def\@this@crossref@end{#7:end}%
4058   \printlineendnote{#1}{#4}%
4059   \IfStrEq{#6}{R}{\ledRcol@false}{ }%
4060   \undef\@this@crossref@start%
4061   \undef\@this@crossref@end%
4062   \nottoggle{Xendlemmadisablefontselection@#4}%
4063     {\select@lemmafонт#1|\csuse{Xendlemmafонт@#4}#2}%
4064     {\csuse{Xendlemmafонт@#4}#2}%
4065   \ifboolexpr{
4066     togl {nosep@}%
4067     or test{\ifcsempy{Xendlemmaseparator@#4}}%
4068   }%

```



```

4069     {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
4070     {\nobreak%
4071     \hskip\csuse{Xendbeforelemmaseparator@#4}}%
4072     \csuse{Xendlemmaseparator@#4}}%
4073     \hskip\csuse{Xendafterlemmaseparator@#4}}%
4074     }%
4075     #3%
4076     \nottoggle{Xendparagraph@#4}{\par}{}%
4077     \def\do##1{%
4078     \setkeys[mac]{falsefootnoteoption}{##1}}%
4079     }%
4080     \notblank{#5}{\docsvlist{#5}}{}}%
4081 }}%
4082
4083 \let\l@d@section=\@gobble
4084
4085 %

```

`\printlineendnote` This macro controls, in endnote, whether the line number is printed or not, according to the series options. Its first argument is the information about lines; its second is the series of the footnote.

```

4086 \newcommand{\printlineendnote}[2]{%
4087   \l@dp@ersefootspec#1|%
4088   \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
4089     \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4090     \l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
4091     \l@dparsedendsub}}%
4092     }%
4093     {\%
4094     \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4095     \l@dparsedstartsub}}%
4096     }%
4097     \ifboolexpr{%
4098     togl {nonum@}%
4099     or togl {Xendnonumber@#2}}%
4100     {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
4101     {%
4102     \iftoggle{Xendnumberonlyfirstinline@#2}%
4103     {\ifcsdef{prevendline#2}%
4104     {\ifcsequal{prevendline#2}{\lineinfo@}%
4105     {\%
4106     \csuse{Xendbhookinplaceofnumber@#2}}%
4107     \ifcsequal{Xendsymnum@#2}%
4108     {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
4109     {\printlineendnotearea{#2}}}%
4110     \csuse{Xendahookinplaceofnumber@2}}%
4111     }%
4112     {\printlineendnotearea{#1}{#2}}}%

```

```

4111     {\printlineendnotearea{#1}{#2}}%
4112     }%
4113     {\printlineendnotearea{#1}{#2}}%We keep every time line
4114     \csxdef{prevendline#2}{\lineinfo@}%
4115     }%
4116 }%
4117 %

```

```

\printsymlineendnotearea 18 \newcommand{\printsymlineendnotearea}[1]{%
4119     \hspace{\csuse{Xendbeforesymlinenum@#1}}%
4120     \csuse{Xendnotenumfont@#1}%
4121     \ifdimequal{\csuse{Xendboxsymlinenum@#1}}{\z@}%
4122     {\csuse{Xendsymlinenum@#1}}%
4123     {\hbox to \csuse{Xendboxsymlinenum@#1}%
4124       {\csuse{Xendsymlinenum@#1}\hfill}}%
4125     }%
4126     \hspace{\csuse{Xendaftersymlinenum@#1}}%
4127 }%
4128 %

```

`\printlineendnotearea` This macro prints the space before the line number, changes the font, then prints the line number and the space after it. It is called by `\endprint` depending of the options about repeating line numbers. The first argument is line information, the second is the notes series (A, B, C, etc.)

```

4129 \newcommand{\printlineendnotearea}[2]{%
4130     \csuse{Xendbhooklinenumber@#2}%
4131     \hspace{\csuse{Xendbeforenumber@#2}}%
4132     \bgroup%
4133     \csuse{Xendnotenumfont@#2}%
4134     \ifdimequal{\csuse{Xendboxlinenum@#2}}{\Opt}%
4135     {\printendlines#1||\ifledRcol@\@Rlineflag\fi}%
4136     {\leavevmode%
4137       \hbox to \csuse{Xendboxlinenum@#2}%
4138       {%
4139         \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}}%
4140         \printendlines#1||\ifledRcol@\@Rlineflag\fi%
4141         \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}}%
4142       }%
4143     \egroup%
4144     \hspace{\csuse{Xendafternumber@#2}}%
4145     \csuse{Xendahooklinenumber@#2}%
4146 }%
4147 %

```

`\doendnotes` `\doendnotes` is the command you use to print one series of endnotes; it takes one argument: the series letter of the note series you want to print. `\Xendinsertsep@` is set to true at the first note of the series, and to false at the last one.

```

4148 \newif\ifXendinsertsep%
4149 \newcommand*\doendnotes}[1]{%
4150   \l@dend@close{#1}%
4151   \begingroup
4152     \makeatletter
4153     \expandafter\let\csname #1end\endcsname=\endprint
4154     \input\jobname.#1end%
4155     \global\Xendinsertsep@false%
4156   \endgroup}
4157 %

```

`\doendnotesbysection` `\doendnotesbysection` is a variant of the previous macro. While `\doendnotes` print endnotes for all of numbered sections `\doendnotesbysection` print the endnotes for the first numbered section at its first call for a series, then for the second section at its second call for the same series, then for the third section at its third call for the same series, and so on.

```

4158 \newcommand*\doendnotesbysection}[1]{%
4159   \l@dend@close{#1}%
4160   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
4161   \begingroup%
4162     \makeatletter%
4163     \def\l@d@section##1{%
4164       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
4165         {\cslet{#1end}{\endprint}}%
4166         {\cslet{#1end}{\@gobblefive}}%
4167     }%
4168     \input\jobname.#1end%
4169     \global\Xendinsertsep@false%
4170   \endgroup%
4171 }%
4172 %

```

We close now the conditional period, which depends on `\ifnoend@`, because the following commands can be used by other commands than those specific to endnotes.

```

4173 }%
4174 %

```

`\setprintendlines` The `\printendlines` macro is similar to `\printlines` but is for printing endnotes rather than footnotes.

The principal difference between foot- and endnotes is that footnotes are printed on the page where they are specified but endnotes are printed at a different point in the document. We need an indication of the source of an endnote; `\setprintendlines` provides this by always printing the page number. The coding is slightly simpler than `\setprintlines`.

First of all, we print the second page number only if the ending page number is different from the starting page number.

```

4175 \newcommand*\setprintendlines}[6]{%
4176   \l@dpnumfalse \l@ddashfalse
4177   \ifnum#4=#1 \else
4178     \l@dpnumtrue
4179     \l@ddashtrue
4180   \fi
4181 %

```

We print the ending line number if: (1) we are printing the ending page number, or (2) it is different from the starting line number.

```

4182   \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
4183   \ifnum#2=#5 \else
4184     \l@d@elintrue
4185     \l@d@dashtrue
4186   \fi
4187 %

```

We print the starting sub-line if it is nonzero.

```

4188   \l@d@ssubfalse
4189   \ifnum#3=0 \else
4190     \l@d@ssubtrue
4191   \fi
4192 %

```

We print the ending sub-line if it is nonzero and: (1) it is different from the starting sub-line number, or (2) the ending line number is being printed.

```

4193   \l@d@eslfalse
4194   \ifnum#6=0 \else
4195     \ifnum#6=#3
4196       \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
4197     \else
4198       \l@d@esltrue
4199       \l@d@dashtrue
4200     \fi
4201   \fi%
4202 %

```

```

4203   \ifl@d@dash%
4204     \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xendtwolines@}\@currentseries}}%
4205     {%
4206     {%
4207       \setistwofollowinglines{#1}{#2}{#4}{#5}%
4208       \ifboolexpr{%
4209         (%
4210           togl {Xendtwolinesbutnotmore@\@currentseries}%
4211           and not%
4212           (%
4213             bool {istwofollowinglines@}%

```

```

4214         )%
4215     )%
4216     or%
4217     (%
4218         (not test{\ifnumequal{#1}{#4}})%
4219         and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
4220     )%
4221 }%
4222 {}%
4223 {%
4224     \l@d@dashfalse%
4225     \l@d@xtwolinestrue%
4226     \l@d@elinfalse%
4227     \l@d@eslfalse%
4228     \ifcseempty{Xendmoreethantwolines@\@currentseries}%
4229     {%
4230         {\ifistwofollowinglines@\else%
4231             \l@d@Xmoreethantwolinestrue%
4232         }%
4233     }%
4234 }%
4235 }%
4236 \fi%
4237 %

```

End of \setprintendlines.

```

4238 }%
4239 %

```

`\printendlines` Now we are ready to print it all.

```

4240 \def\printendlines#1|#2|#3|#4|#5|#6|#7|#8|{\begingroup
4241     \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
4242 %

```

The only subtlety left here is when to print a period between numbers. But the only instance in which this is tricky is for the ending sub-line number: it could be coming after the starting sub-line number (in which case we want only the dash) or after an ending line number (in which case we need to insert a period).

So, first, print the start lines.

```

4243     \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
4244     {\bgroup}%
4245     {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%
4246     \wrap@edcrossref{\@this@crossref@start}{\printnnum{#1}}%
4247     \ifl@d@dash%
4248     \ifl@d@pnum%
4249     \csuse{Xendlineprefixsingle@\@currentseries}%
4250     \else%

```

```

4251 \ifcsemtyp{Xendlineprefixmore@\@currentseries}%
4252   {\csuse{Xendlineprefixsingle@\@currentseries}}
4253   {\csuse{Xendlineprefixmore@\@currentseries}}%
4254 \fi%
4255 \else%
4256   \csuse{Xendlineprefixsingle@\@currentseries}%
4257 \fi%
4258 \wrap@edcrossref{\@this@crossref@start}{\linenumrep{#2}}%
4259 \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
4260 \ifl@d@ssub%
4261   \csuse{Xendsublinesep@\@currentseries}%
4262   \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
4263 \fi%
4264 \egroup%
4265 %

```

And now, print the dash + the end line number, or the line number range symbol.

```

4266 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
4267   {\bgroup}%
4268   {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
4269 \ifl@d@Xtwolines%
4270   \ifl@d@Xmorethantwolines%
4271     \csuse{Xendmorethantwolines@\@currentseries}%
4272   \else%
4273     \csuse{Xendtwolines@\@currentseries}%
4274   \fi%
4275 \else%
4276   \ifl@d@dash%
4277     \ifdefined\linerangesep%
4278       \linerangesep%
4279     \else%
4280       \csuse{Xendlinerangeseparator@\@currentseries}%
4281     \fi%
4282   \fi%
4283   \ifl@d@pnum%
4284     \wrap@edcrossref{\@this@crossref@end}\printnpnum{#4}%
4285   \fi%
4286   \ifl@d@elin%
4287     \ifl@d@pnum\csuse{Xendlineprefixsingle@\@currentseries}\fi%
4288     \wrap@edcrossref{\@this@crossref@end}{\linenumrep{#5}}%
4289     \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
4290   \fi%
4291   \ifl@d@esl%
4292     \ifl@d@elin%
4293       \csuse{Xendsublinesep@\@currentseries}%
4294     \fi%
4295     \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}%
4296   \fi%
4297 \fi%
4298 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%

```

```

4299     {}%
4300     {\hfill}%Prevent underfull hbox
4301   \egroup%
4302   \endgroup%
4303 }%
4304
4305 %

```

`\printnpnum` A macro to print a page number in an endnote. Should not be override anymore

```

4306 \newcommand*\printnpnum[1]{\csuse{Xendbeforepagenumber@ \@currentseries
4307 }#1\csuse{Xendafterpagenumber@ \@currentseries}}
4308 %

```

## XX Generate series of notes

In this section, X means the name of the series (A, B etc.)

`\series` `\series` creates one more new series. It is a public command, which just loops on the private command `\newseries@`.

```

4309 \newcommand{\newseries}[1]{%
4310   \def\do##1{\newseries@{##1}}%
4311   \docsvlist{#1}
4312 }
4313 %

```

`\@series` The `\series@` macro is an etoolbox list, which contains the name of all series.

```

4314 \newcommand{\@series}{}
4315 %

```

The command `\newseries@``\series` creates a new series of the footnote.

```

\newseries@16 \newcommand{\newseries@}[1]{
4317 %

```

### XX.1 Test if series is still existing

```

4318   \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
4319   {%
4320 %

```

### XX.2 Init specific to reledpar

When calling `\newseries@` after having loaded `reledpar`, we need to load specific setting.

```

4321 \ifdefined\newseries@par%
4322 \newseries@par{#1}%
4323 \fi%
4324 %

```

### XX.3 For critical footnotes

Critical footnotes are those which start with letters. We look for the `\nocritical` option of `reledmac`.

```

4325 \unless\ifnocritical@
4326 %

```

#### XX.3.1 Options

```

4327 \newtoggle{Xlineflag@#1}
4328 \newtoggle{Xparindent@#1}
4329 \newtoggle{Xlemmadisablefontselection@#1}
4330 \csgdef{Xhangindent@#1}{Opt}%
4331 \csgdef{Xragged@#1}{}%
4332 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
4333 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
4334 \csgdef{Xcolalign@#1}{\raggedright}%
4335 \csgdef{Xnotenumfont@#1}{\normalfont}%
4336 \csgdef{Xnotefontsize@#1}{\footnotesize}%
4337 \csgdef{Xbhooknote@#1}{}%
4338 \csgdef{Xbhookgroup@#1}{}%
4339
4340 \csgdef{Xboxlinenum@#1}{Opt}%
4341 \csgdef{Xboxlinenumalign@#1}{L}%
4342
4343 \csgdef{Xboxstartlinenum@#1}{Opt}%
4344 \csgdef{Xboxendlinenum@#1}{Opt}%
4345
4346 \csgdef{Xboxsymlinenum@#1}{Opt}%
4347 \newtoggle{Xnumberonlyfirstinline@#1}%
4348 \newtoggle{Xnumberonlyfirstintwolines@#1}%
4349 \csgdef{Xtwolines@#1}{}%
4350 \csgdef{Xmorethantwolines@#1}{}%
4351 \csgdef{Xsublinesep@#1}{\fullstop}%
4352 \newtoggle{Xtwolinesbutnotmore@#1}%
4353 \newtoggle{Xtwolinesonlyinsamepage@#1}%
4354 \newtoggle{Xonlypstart@#1}%
4355 \newtoggle{Xpstarteverytime@#1}%
4356 \newtoggle{Xpstart@#1}%
4357 \newtoggle{Xstanza@#1}%
4358 \csgdef{Xstanzaseparator@#1}{}%
4359 \csgdef{Xsymlinenum@#1}{}%

```



```

4360 \newtoggle{Xnonnumber@#1}%
4361 \csgdef{Xbeforenumber@#1}{Opt}%
4362 \csgdef{Xtxtbeforenumber@#1}{}%
4363 \csgdef{Xafternumber@#1}{0.5em}%
4364 \newtoggle{Xnonbreakableafternumber@#1}%
4365 \csgdef{Xbeforenumber@#1}{\csuse{Xbeforenumber@#1}}%
4366 \csgdef{Xaftersymnumber@#1}{\csuse{Xafternumber@#1}}%
4367 \csgdef{Xinplaceofnumber@#1}{1em}%
4368 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
4369 \csgdef{Xbeforelemmaseparator@#1}{0em}%
4370 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
4371 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
4372 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}
4373 \csgdef{Xafterrule@#1}{Opt}
4374 \csgdef{Xtxtbeforenotes@#1}{%
4375 \csgdef{Xmaxhnotes@#1}{0.8\vsz}
4376 \newtoggle{Xnoteswidthliketwocolumns@#1}%
4377 \csgdef{Xparafootsep@#1}{%
4378 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}
4379 \csgdef{Xlinerangeseparator@#1}{\endashchar}%
4380
4381 \csgdef{Xlemmafont@#1}{%
4382 \csgdef{Xwidth@#1}{\hsz}
4383 %

```

### XX.3.2 Create inserts, needed to add notes in foot

As regards inserts, see chapter 15 of *The TeXbook* by D. Knuth.

```

4384 \expandafter\newinsert\csname #1footins\endcsname%
4385 \unless\ifnoledgroup%
4386 \expandafter\newinsert\csname mp#1footins\endcsname%
4387 \fi%
4388 %

```

### XX.3.3 Create commands for critical apparatus, \Afootnote, \Bfootnote etc.

Note the double # in command: it is because command it is made inside another command.

```

4389 \global\newbool{parapparatus@}{\expandafter\newcommand\expandafter
*}{\expandafter\newcommand}\csname #1footnote\endcsname [2] [] {%
4390 \ifnum\@edtext@level>0%
4391 \begingroup%
4392 \newcommand{\content}{##2}%
4393 \ifnumberedpar%
4394 \ifledRcol%
4395 \ifluatex%
4396 \footnotelang@lua[R]%
4397 \fi%
4398 \@ifundefined{xpg@main@language}%if polyglossia

```

```

4399     {}%
4400     {\footnotelang@poly[R]}%
4401 \footnoteoptions@{R}{#1}{true}%
4402 \xright@appenditem{%
4403   \ifbool{indtl@innote}%
4404     {\unexpanded{\let\index\nindex}}%
4405     {}%
4406   \ifbool{indtl@notenumber}%
4407     {\unexpanded{\let\index\nindex}}%There is no note
...number so
4408     {}%
4409     \noexpand\Xnote@true%
4410     \noexpand\prepare@preXnotes{#1}%
4411     \noexpand\prepare@edindex@fornote{\l@d@nums}%
4412     \unexpanded{\def\sw@list@inedtext{\expandafter\
unexpanded\expandafter{\sw@inthisedtext}}%The value of the \sw@inthisedtext
of current \edtext will be pushed to \sw@list@inedtext when the notes are
expanded.
4413     \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4414     \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
4415     \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
4416     \noexpand\csuse{v#1footnote}{#1}%
4417     {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4418     \noexpand\Xnote@false%
4419     \unexpanded{\undef\@this@crossref@start}%
4420     \unexpanded{\undef\@this@crossref@end}%
4421     \ifbool{indtl@innote}%
4422       {\unexpanded{\let\index\orig@@index}}%
4423       {}%
4424     \ifbool{indtl@notenumber}%
4425       {\unexpanded{\let\index\orig@@index}}%
4426       {}%
4427   }\to\inserts@listR
4428   \footnoteoptions@{R}{#1}{false}%
4429   \global\advance\insert@countR \@ne%
4430 \else%
4431   \ifluatex%
4432     \footnotelang@lua%
4433   \fi%
4434   \@ifundefined{xpg@main@language}%if polyglossia
4435     {}%
4436     {\footnotelang@poly}%
4437   \footnoteoptions@{L}{#1}{true}%
4438   \xright@appenditem{%
4439     \ifbool{indtl@innote}%
4440       {\unexpanded{\let\index\nindex}}%
4441       {}%

```

```

4442         \ifbool{indtl@notenumber}%
4443             {\unexpanded{\let\index\nindex}}%There is no note
...number so
4444             {}%
4445         \noexpand\Xnote@true%
4446         \noexpand\prepare@preXnotes{#1}%
4447         \noexpand\prepare@edindex@fornote{\l@d@nums}%
4448         \unexpanded{\def\sw@list@inedtext}{\expandafter\
unexpanded\expandafter{\sw@inthisedtext}}%The value of the \sw@inthisedtext
of current edtext will be pushed to \sw@list@inedtext when the notes are
expanded.
4449         \ifl@dpairing%
4450             \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4451         \fi%
4452         \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
4453         \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
4454         \noexpand\csuse{v#1footnote}%
4455             {#1}%
4456             {\l@d@nums}{\expandonce\@tag}{\expandonce\content
}}%
4457         \unexpanded{\undef\@this@crossref@start}%
4458         \unexpanded{\undef\@this@crossref@end}%
4459         \noexpand\Xnote@false%
4460         \ifbool{indtl@innote}%
4461             {\unexpanded{\let\index\orig@@index}}%
4462             {}%
4463         \ifbool{indtl@notenumber}%
4464             {\unexpanded{\let\index\orig@@index}}%
4465             {}%
4466         }\to\inserts@list
4467         \global\advance\insert@count \@ne%
4468         \footnoteoptions@{L}{#1}{false}%
4469         \fi
4470     \else
4471         \csuse{v#1footnote}{#1}{0|0|0|0|0|0|0|0}{#1}%
4472         \fi%
4473     \endgroup%
4474 \else%
4475     \led@err@FootnoteWithoutEdtext%
4476     \fi%
4477 \ignorespaces%
4478 }
4479 %

```

We need to be able to modify reledmac's footnote macros and restore their

```

4480     \global\csletcs{#1@@footnote}{#1footnote}
4481 %

```

### XX.3.4 Set standard display

```
4482 \Xarrangement@normal{#1}%
4483 %
```

End of for critical footnotes.

```
4484 \fi
4485 %
```

## XX.4 For familiar footnotes

Familiar footnotes are those which end with letters. We look for the `nofamiliar` option of `reledmac`.

```
4486 \unless\ifnofamiliar@
4487 %
```

### XX.4.1 Options

```
4488 \newtoggle{parindentX@#1}
4489 \csgdef{hangindentX@#1}{Opt}%
4490 \csgdef{raggedX@#1}{}%
4491 \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
4492 \csgdef{hsizethreecolX@#1}{.3 \hsize}%
4493 \csgdef{colalignX@#1}{\raggedright}%
4494 \csgdef{notenumfontX@#1}{\normalfont}%
4495 \csgdef{notefontsizeX@#1}{\footnotesize}%
4496 \csgdef{bhooknoteX@#1}{}%
4497 \csgdef{bhookgroupX@#1}{}%
4498 \csgdef{afterruleX@#1}{Opt}
4499 \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
4500 \csgdef{maxhnotesX@#1}{0.8\vsizex}%
4501 \newtoggle{noteswidthliketwocolumnsX@#1}%
4502 \csgdef{parafootsepX@#1}{}%
4503 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4504 \csgdef{widthX@#1}{\hsizex}%
4505 % End of for familiar footnotes.
4506 % \subsubsection{Create inserts, needed to add notes in foot}
4507 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4508 % \begin{macrocode}
4509 \expandafter\newinsert\csname footins#1\endcsname%
4510 \unless\ifnoledgroup%
4511 \expandafter\newinsert\csname mpfootins#1\endcsname%
4512 \fi%
4513 %
```

### XX.4.2 Create tools for familiar footnotes (`\footnoteX`)

First, create the `\footnoteX` command. Note the double # in command: it is because a command is called inside another command.

```

4514 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4515 \begingroup%
4516 \prepare@prenotesX{#1}%
4517 \newcommand{\content}{##1}%
4518 %
4519 %

```

If we are preparing parallel typesetting, we cannot just increase the footnote counter. Read `reledpar`'s handbook about that (V.2.1 p. 46).

```

4520 \global\expandafter\advance\csname footnote#1@reading\
endcsname by \@ne%
4521 \ifl@dpairing%
4522 \ifcsdef{footnote#1reading\the\csname footnote#1@reading\
endcsname=typeset}%
4523 {\setcounter{footnote#1}{\csuse{footnote#1reading\the\
csname footnote#1@reading\endcsname=typeset}}}%
4524 {\setcounter{footnote#1}{\the\csname footnote#1@reading
\endcsname}}}%
4525 \else%
4526 \stepcounter{footnote#1}%
4527 \fi%
4528 %

```

And now, the feature not depending of whether we are preparing parallel typesetting

```

4529 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4530 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4531 {\csuse{@footnotemark#1}}%
4532 {}%
4533 \ifluatex%
4534 \xdef\footnote@luatextextdir{\the\textdir}%
4535 \xdef\footnote@luatexpardir{\the\pardir}%
4536 \fi%
4537 \if@ledgroup%
4538 \led@set@index@fornote{#1}%
4539 \fi%
4540 \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
4541 \ifbool{indtl@innote}%
4542 {\let\index\orig@@index}%
4543 {}%
4544 \ifbool{indtl@notenumber}%
4545 {\let\index\orig@@index}%
4546 {}%
4547 \endgroup%
4548 }
4549 %

```

Then define the counters. The  $\text{\LaTeX}$  counter `footnoteX` is the only one manipulated by the user. This is the one which is printed. The  $\text{\TeX}$  counter `\footnoteX@reading`

is increased at each footnote. It is used for hyperlinks, for using `hyperlink` package, and for getting the correct footnote number when using parallel typesetting (V.2.1 p. 46).

```

4550     \newcounter{footnote#1}
4551     \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4552     \expandafter\newcount\csname footnote#1@reading\endcsname%
4553 %

```

Do not forget to initialize series

```

4554     \arrangementX@normal{#1}%
4555     \fi
4556 %

```

## XX.5 The endnotes

Endnotes are commands like `\Xendnote`, where X is a series letter. First, we check for the `noend` options.

```

4557     \unless\ifnoend@
4558 %

```

### XX.5.1 The auxiliary file

`\l@d@Xend` Endnotes of all varieties are saved up in a file, one by series, typically named `\jobname.Xend`.  
`\ifl@dend@X` `\l@d@end` is the output stream number for this file, and `\ifl@dend@X` is a flag that is  
`\l@dend@Xtrue` true when the file is open.  
`\l@dend@Xfalse`

```

4559     \expandafter\newwrite\csname l@d@#1end\endcsname%
4560     \expandafter\newif\csname ifl@dend@#1\endcsname%
4561 %

```

### XX.5.2 The main macro

The `\Xendnote` macro functions to write one endnote to the `.Xend` file. We change `\newlinechar` so that in the file every space becomes the start of a new line; this generally ensures that a long note does not exceed restrictions on the length of lines in files.

```

4562     \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
4563 usedefault]{%
4564     \bgroup%
4565     \newlinechar='40%
4566     \global\@noneed@Footnotetrue%
4567     \newcommand{\content}{##2}%
4568     \stepcounter{labidx}%
4569     \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%

```

```

4570 \expandafter\string\csname #1end\endcsname%
4571 {\ifnumberedpar@l@d@nums\fi}%
4572 {\ifnumberedpar@expandonce@tag\fi}%
4573 {\expandonce\content}%
4574 {#1}%
4575 {\unexpanded{##1}}%
4576 {\ifledRcol R\else L\fi}%
4577 {\theadtext}%
4578 \@percentchar%
4579 }%
4580 \egroup%
4581 \ignorespaces%
4582 }%
4583 %

```

\Xendnote commands called \Xend commands on to the endnote file; these are analogous to the various footfmt commands above, and they take the same arguments. When we process this file, we want to pick out the notes of one series and ignore all the rest. To do that, we equate the end command for the series we want to \endprint, and leave the rest equated to \@gobblefive, which just skips over its five arguments.

```

4584 \global\cslet{#1end}{\@gobblefive}
4585 %
4586 %

```

We need to store the number of times \doendnotesbysection is called for one series.

```

4587 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4588 %

```

### XX.5.3 The options

```

4589 \csgdef{Xendtwolines@#1}{}%
4590 \csgdef{Xendmoreethantwolines@#1}{}%
4591 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4592 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4593 \newtoggle{Xendlemmadisablefontselection@#1}{}%
4594 \csgdef{Xendnotenumfont@#1}{\normalfont}%
4595 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4596 \csgdef{Xendbhooknote@#1}{}%
4597
4598 \csgdef{Xendsublinesep@#1}{\fullstop}%
4599
4600 \csgdef{Xendbeforenumber@#1}{Opt}
4601 \csgdef{Xendafternumber@#1}{0.5em}
4602
4603 \csgdef{Xendboxlinenum@#1}{Opt}%
4604 \csgdef{Xendboxlinenumalign@#1}{L}%
4605
4606 \csgdef{Xendboxstartlinenum@#1}{Opt}%

```

```

4607 \csgdef{Xendboxendlinenum@#1}{Opt}%
4608
4609 \csgdef{Xendlemmaseparator@#1}{}%
4610 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4611 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4612 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4613
4614 \newtoggle{Xendparagraph@#1}%
4615 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4616 \csgdef{Xendsep@#1}{}%
4617
4618 \csgdef{Xendinplaceofnumber@#1}{Opt}%
4619 \newtoggle{Xendnonnumber@#1}%
4620
4621 \csgdef{Xendhangindent@#1}{Opt}%
4622 \newtoggle{Xendnumberonlyfirstinline@#1}%
4623 \newtoggle{Xendnumberonlyfirstintwolines@#1}%
4624
4625 \csgdef{Xendbeforesymmlinum@#1}{\csuse{Xendbeforenumber@#1}}%
4626 \csgdef{Xendaftersymmlinum@#1}{\csuse{Xendafternumber@#1}}%
4627 \csgdef{Xendsymmlinum@#1}{}%
4628 \csgdef{Xendboxsymmlinum@#1}{Opt}%
4629
4630 \csgdef{Xendbhooklinenumber@#1}{}%
4631 \csgdef{Xendehooklinenumber@#1}{}%
4632 \csgdef{Xendbhookinplaceofnumber@#1}{}%
4633 \csgdef{Xendehookinplaceofnumber@#1}{}%
4634
4635 \csgdef{Xendlinerangeseparator@#1}{\endashchar}%
4636
4637 \csgdef{Xendbeforepagenumber@#1}{p.}%
4638 \csgdef{Xendafterpagenumber@#1}{) }%
4639 \csgdef{Xendlineprefixsingle@#1}{}%
4640 \csgdef{Xendlineprefixmore@#1}{}%
4641
4642 \newtoggle{Xendlineflag@#1}
4643
4644 \csgdef{Xendlemmafont@#1}{}%
4645 %

```

End of endnotes declaration

```

4646 \fi%
4647 %

```

Dump series in \@series

```

4648 \listxadd{\@series}{#1}
4649 }
4650 }% End of \newseries
4651 %

```



## XX.6 Init standards series (A,B,C,D,E)

```
4652 \expandafter\newseries\expandafter{\default@series}
4653 %
```

## XXI Setting series display

### XXI.1 Change series order

`\seriesatbegin` `\seriesatbegin{<s>}` changes the order of series, to put the series `<s>` at the beginning of the list. The series can be the result of a command.

```
4654 \newcommand{\seriesatbegin}[1]{%
4655   \StrDel{\@series}{#1}[\@series]%
4656   \edef\@new{}%
4657   \listadd{\@new}{#1}%
4658   \listadd{\@new}{\@series}%
4659   \xdef\@series{\@new}%
4660 }
4661 %
```

`\seriesatend` And `\seriesatend` moves the series to the end of the list.

```
4662 \newcommand{\seriesatend}[1]{%
4663   \StrDel{\@series}{#1}[\@series]%
4664   \edef\@new{}%
4665   \listadd{\@new}{\@series}%
4666   \listadd{\@new}{#1}%
4667   \xdef\@series{\@new}%
4668 }
4669 %
```

### XXI.2 Test series order

`\ifseriesbefore` `\ifseriesbefore{<seriesA>}{<seriesB>}{<>true>}{<>false>}` expands `<>true>` if `<seriesA>` is printed before `<seriesB>`, expands `<>false>` otherwise.

```
4670 \newcommand{\ifseriesbefore}[4]{%
4671   \StrPosition{\@series}{#1}[\@first]%
4672   \StrPosition{\@series}{#2}[\@second]%
4673   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4674 }
4675 %
```

#### XXI.2.1 Get the first series

In some specific case, we need to know the first series of the list of series.

```

\@getfirstseries 76 \newcommand{\@getfirstseries}{%
4677 \ifdefempty{\@series}%
4678 {\xdef\@firstseries{}}%
4679 {\StrChar{\@series}{1}[\@firstseries]}%
4680 }%
4681 %

```

## XXI.3 Series setting

### XXI.3.1 General way of working

The setting's command (like `\numberonlyfirstinline`), also called “hooks” can be divided in two categories: those which require a string values and those which require a boolean value. The first category includes those which require a length value, because we store the length's expression send by user and we evaluate it only in the commands which requires to know the setting. The second category require boolean value only when it is set to FALSE. Otherwise, we understand the insinuated value is TRUE.

For each “hook” command, we store the value in commands (first category) or a `etoolbox`'s toggle (second category) which names are in the form `\<hook>@<series>`. For example when calling `\twolines{<sq.>}`, we store `sq.` in commands `\twolines@A`, `\twolines@B`, `\twolines@C`...for each series defined for use with `reledmac`, or, if the [`<series>`] optional argument was send, for each series of this argument.

These values are tested in some specific places, scattered in all the code, depending of their effects. The default values are defined by the `\newseries@` command.

In order to prevent code duplication, we have created some generic commands. Some of them change the value of any hook send as argument. Some other, getting a hook name, generate the user level commands.

### XXI.3.2 Tools to set options

`\settoggle@series` `\settoggle@series{<series>}{<toggle>}{<value>}` is a generic command to switch toggles for some series. The arguments are:

- #1 (mandatory): the series for which the hooks should be set. If empty, all the series will be affected.
- #2 (mandatory): the name of the hook.
- #3 (mandatory): the new value of toggle (true or false).
- #4 (optional): if equal to `reload`, reload the footnote setting (call again `\Xarrangement` or `\arrangementX` or ... depending of the footnote display).
- #5 (optional): if not empty, and if #1 is empty, change the hook setting for pseudo-series, as `appref`.

```

4682 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4683   \def\do##1{%
4684     \global\settoggle{#2@##1}{#3}%
4685     \ifstrequal{#4}{critical}{
4686       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
4687     }{}
4688     \ifstrequal{#4}{familiar}{
4689       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
4690     }{}
4691   }%
4692   \ifstreempty{#1}{%
4693     \dolistloop{\@series}%
4694     \ifstreempty{#5}{}%
4695       \docsvlist{#5}%
4696     }
4697   }%
4698   {%
4699     \docsvlist{#1}%
4700   }%
4701 }
4702 %

```

`\setcommand@series` `\setcommand@series{<series>}{<command>}{<value>}` is a generic command to store hook's value into commands specific to some series. The arguments are:

- #1 (mandatory): the series for which the hooks should be set. If empty, all the series will be affected.
- #2 (mandatory): the name of the hook.
- #3 (mandatory): the new value of the hook/command.
- #4 (optional): if equal to `reload`, reload the footnote setting (call `\footnormal` or `\footparagraph` or ... depending of the footnote display).
- #5 (optional): if not empty, and if #1 is empty, change the hook setting for pseudo-series, as `appref`.

```

4703 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4704   \def\do##1{
4705     \csgdef{#2@##1}{#3}
4706     \ifstrequal{#4}{critical}{%
4707       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
4708     }{}
4709     \ifstrequal{#4}{familiar}{%
4710       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
4711     }{}%
4712   }%
4713   \ifstreempty{#1}{%
4714     \dolistloop{\@series}%

```



```

4747 }%
4748 \docsvlist{#2}%
4749 }%
4750 }
4751 %

```

`\newhooktoggle@series@reload` `\newhookcommand@toggle@reload` does the same thing as `\newhooktoggle@series` but the commands created by this macro also reload the series arrangement, depending of type os notes

```

4752 \newcommand{\newhooktoggle@series@reload}[2]{%
4753 \global\expandafter\newcommandx\expandafter*\csname #1\endcsname [2] [1,2={
true},usedefault]{%
4754 \settoggle@series{##1}{#1}{##2}[#2]%
4755 }%
4756 }%
4757 %

```

`\newhookcommand@series@reload` `\newhookcommand@series@reload` does the same thing as `\newhookcommand@series` but the commands created by this macro also reload the series' arrangement.

```

4758 \newcommand{\newhookcommand@series@reload}[2]{%
4759 \global\expandafter\newcommand\expandafter*\csname #1\endcsname [2] []{%
4760 \setcommand@series{##1}{#1}{##2}[#2]%
4761 }%
4762 }
4763 %

```

### XXI.3.4 Options for critical notes

Before generating the commands that are used to set the critical notes, such as `\Xnumberonlyfirstinline`, `\Xlemmaseparator` and the like, we check the `nocritical` option.

```

4764 \unless\ifnocritical@
4765 \newhookcommand@series{Xlemmafont}%
4766 \newhooktoggle@series{Xparindent}
4767 \newhookcommand@series{Xhangindent}
4768 \newhookcommand@series{Xragged}
4769 \newhookcommand@series{Xhsizetwocol}
4770 \newhookcommand@series{Xhsizethreecol}
4771 \newhookcommand@series{Xcolalign}%
4772 \newhookcommand@series{Xnotenumfont}
4773 \newhookcommand@series{Xbhooknote}
4774 \newhookcommand@series@reload{Xbhookgroup}{critical}
4775 \newhookcommand@series{Xboxsymlinenum}%
4776 \newhookcommand@series{Xsymlinenum}
4777 \newhookcommand@series{Xbeforenumber}
4778 \newhookcommand@series{Xtxtbeforenumber}

```

```

4779 \newhookcommand@series{Xafternumber}
4780 \newhookcommand@series{Xbeforesymmlinenum}
4781 \newhookcommand@series{Xaftersymmlinenum}
4782 \newhookcommand@series{Xinplaceofnumber}
4783 \newhookcommand@series{Xlemmaseparator}
4784 \newhookcommand@series{Xbeforelemmaseparator}
4785 \newhookcommand@series{Xafterlemmaseparator}
4786 \newhookcommand@series{Xinplaceoflemmaseparator}
4787 \newhookcommand@series{Xtxtbeforenotes}
4788 \newhookcommand@series@reload{Xafterrule}{critical}
4789 \newhooktoggle@series{Xnumberonlyfirstinline}
4790 \newhooktoggle@series{Xnumberonlyfirstintwolines}
4791 \newhooktoggle@series{Xnonumber}
4792 \newhooktoggle@series{Xpstart}
4793 \newhooktoggle@series{Xpstarteverytime}%
4794
4795 \newhooktoggle@series{Xstanza}%
4796 \newhookcommand@series{Xstanzaseparator}%
4797
4798 \newhooktoggle@series{Xonlypstart}
4799 \newhooktoggle@series{Xnonbreakableafternumber}
4800 \newhooktoggle@series{Xlemmadisablefontselection}
4801 \newhookcommand@series@reload{Xmaxhnotes}{critical}
4802 \newhookcommand@series@reload{Xbeforenotes}{critical}
4803 \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
4804 \newhookcommand@series@reload{Xnotefontsize}{critical}
4805
4806 \newhookcommand@series{Xboxlinenum}%
4807 \newhookcommand@series{Xboxlinenumalign}%
4808
4809 \newhookcommand@series{Xboxstartlinenum}%
4810 \newhookcommand@series{Xboxendlinenum}%
4811
4812 \newhookcommand@series{Xafternote}%
4813 \newhookcommand@series{Xparafootsep}
4814
4815 \newhookcommand@series@reload{Xwidth}{critical}%
4816
4817 \ifundef{\Xhsize}%
4818   {%
4819     \newcommandx{\Xhsize}[2][1,usedefault]{%
4820       \led@warning@Xhsize@deprecated%
4821       \Xwidth[#1]{#2}%
4822     }%
4823   }%
4824   {}%
4825 \fi
4826 \newhooktoggle@series{Xlineflag}[appref,SEref]
4827 \newhookcommand@series{Xtwolines}[appref,SEref]
4828 \newhookcommand@series{Xmorethantwolines}[appref,SEref]

```

```

4829 \newhookcommand@series{Xsublinesep}[appref,SEref,side]
4830 \newhooktoggle@series{Xtwolinesbutnotmore}[appref,SEref]
4831 \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref,SEref]
4832 \newhookcommand@series{Xlinrangeseparator}[appref,SEref]
4833 %

```

### XXI.3.5 Options for familiar notes

Before generating the optional commands for familiar notes, we check the `\nofamiliar` option.

```

4834 \unless\ifnofamiliar@
4835   \newhooktoggle@series{parindentX}
4836   \newhookcommand@series{hangindentX}
4837   \newhookcommand@series{raggedX}
4838   \newhookcommand@series{hsizetwocolX}
4839   \newhookcommand@series{hsizethreecolX}
4840   \newhookcommand@series{colalignX}%
4841   \newhookcommand@series{notenumfontX}
4842   \newhookcommand@series{bhooknoteX}
4843   \newhookcommand@series@reload{bhookgroupX}{familiar}
4844   \newhookcommand@series@reload{beforenotesX}{familiar}
4845   \newhookcommand@series@reload{maxhnotesX}{familiar}
4846   \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
4847   \newhookcommand@series@reload{afterruleX}{familiar}
4848   \newhookcommand@series@reload{notefontsizeX}{familiar}
4849   \newhookcommand@series{afternoteX}
4850   \newhookcommand@series{parafootsepX}
4851   \newhookcommand@series@reload{widthX}{familiar}%
4852   \ifundef{\hsizeX}%
4853     {%
4854       \newcommandx{\hsizeX}[2][1,usedefault]{%
4855         \led@warning@hsizeX@deprecated%
4856         \widthX[#1]{#2}%
4857       }%
4858     }%
4859   {}%
4860 \fi
4861 %

```

### XXI.3.6 Options for endnotes

Before generating the commands that are used to set the endnotes, such as `\Xnumberonlyfirstinline`, `\Xlemmaseparator+` and the like, we check the `noend` option.

```

4862 \unless\ifnoend@
4863   \newhookcommand@series{Xendnotenumfont}
4864   \newhookcommand@series{Xendlemmafont}%
4865   \newhookcommand@series{Xendbhooknote}

```

```

4866 \newhookcommand@series{Xendboxlinenum}%
4867 \newhookcommand@series{Xendboxlinenumalign}%
4868
4869
4870 \newhookcommand@series{Xendboxstartlinenum}%
4871 \newhookcommand@series{Xendboxendlinenum}%
4872
4873 \newhookcommand@series{Xendnotefontsize}
4874 \newhooktoggle@series{Xendlemmadisablefontselection}
4875 \newhookcommand@series{Xendlemmaseparator}
4876 \newhookcommand@series{Xendbeforelemmaseparator}
4877 \newhookcommand@series{Xendafterlemmaseparator}
4878 \newhookcommand@series{Xendinplaceoflemmaseparator}
4879
4880 \newhookcommand@series{Xendbeforenumber}%
4881 \newhookcommand@series{Xendafternumber}%
4882
4883 \newhooktoggle@series{Xendparagraph}
4884 \newhookcommand@series{Xendafternote}
4885 \newhookcommand@series{Xendsep}
4886
4887 \newhookcommand@series{Xendinplaceofnumber}%
4888 \newhooktoggle@series{Xendnonumber}%
4889
4890 \newhooktoggle@series{Xendnumberonlyfirstinline}%
4891 \newhooktoggle@series{Xendnumberonlyfirstintwolines}%
4892
4893 \newhookcommand@series{Xendsymmlinenum}%
4894 \newhookcommand@series{Xendbeforesymmlinenum}%
4895 \newhookcommand@series{Xendaftersymmlinenum}%
4896 \newhookcommand@series{Xendboxsymmlinenum}%
4897
4898 \newhookcommand@series{Xendbhooklinenumber}%
4899 \newhookcommand@series{Xendahooklinenumber}%
4900 \newhookcommand@series{Xendbhookinplaceofnumber}%
4901 \newhookcommand@series{Xendahookinplaceofnumber}%
4902
4903 \newhookcommand@series{Xendhangindent}%
4904
4905
4906 \fi
4907 \newhooktoggle@series{Xendlineflag}[apprefwithpage,SErefwithpage]
4908 \newhookcommand@series{Xendtwolines}[apprefwithpage,SErefwithpage]
4909 \newhookcommand@series{Xendmoreethantwolines}[apprefwithpage,SErefwithpage]
4910 \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage,SErefwithpage]
4911 \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage,
SErefwithpage]
4912 \newhookcommand@series{Xendlinerangeseparator}[apprefwithpage,SErefwithpage
]

```



```

4913 \newhookcommand@series{Xendbeforepagenumber}[apprefwithpage,SErefwithpage,
SErefonlypage]
4914 \newhookcommand@series{Xendafterpagenumber}[apprefwithpage,SErefwithpage]
4915 \newhookcommand@series{Xendlineprefixsingle}[apprefwithpage,SErefwithpage]
4916 \newhookcommand@series{Xendlineprefixmore}[apprefwithpage,SErefwithpage]
4917 \newhookcommand@series{Xendsublinesep}[apprefwithpage,SErefwithpage]
4918
4919 %

```

## XXI.4 Hooks for a particular footnote

`\newhooktoggle@specific` `\newhooktoggle@specific` is a generic command to create boolean hook specific to a note.

```

4920 \newcommand{\newhooktoggle@specific}[1]{%
4921   \newtoggle{#1}%
4922   \define@key[mac]{truefootnoteoption}{#1}[\global\settoggle{#1}{true}]%
When enabling footnote option
4923   \define@key[mac]{falsefootnoteoption}{#1}[\global\settoggle{#1}{false
}}
4924 }
4925 %

```

`\newhookarg@specific` `\newhookarg@specific` is a generic command to create argumen hook specific to a note.

```

4926 \newcommand{\newhookarg@specific}[1]{%
4927   \define@key[mac]{truefootnoteoption}{#1}{\global\def\linrangesep@{##1}}%
When enabling footnote option
4928   \define@key[mac]{falsefootnoteoption}{#1}{\global\undef\linrangesep@}%
When
4929 }
4930 %

```

And now, we define some hooks specific to a note.

```

4931 \newhooktoggle@specific{fulllines}%
4932 \newhooktoggle@specific{nonum}
4933 \newhooktoggle@specific{nosep}
4934 \newhookarg@specific{linrangesep}
4935 %

```

`linrangesep@` `\linrangesep@` is defined by the option `linrangesep` of critical notes to change temporarily the line range separator for a specific line. As we have to define it before typesetting the line and undefine it after, we use the family of `xkeyval` package's key.

```

4936 %

```

`\nomk@` `\nomk@` toggle is used by `reledpar` to remove the footnote mark in the text when using `\footnoteXmk`. Read `reledpar` handbook.

```
4937 \newtoggle{nomk@}%
4938 %
```

### XXI.5 Alias

`\Xnolemmaseparator` `\Xnolemmaseparator` [*series*] is just an alias for `\Xlemmaseparator` [*series*] {}.

```
4939 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]}
4940 %
```

## XXII Output routine

Now we begin the output routine and associated things.

### XXII.0.1 Page number management

`\pageno` `\pageno` is a page number, starting at 1, and `\advancepageno` increments the number.  
`\advancepageno`

```
4941 \countdef\pageno=0 \pageno=1
4942 \newcommand*{\advancepageno}{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
4943 \else\global\advance\pageno\@ne\fi}
4944
4945 %
```

### XXII.0.2 Extra footnotes output

With luck we might only have to change `\@makecol` and `\@reinserts` of the  $\TeX$ 's kernel. Since `reledmac`, we use `etoolbox`'s patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

`\doxtrafeet` `\doxtrafeet` is the code extending `\@makecol` to cater for the extra `reledmac` feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of  $\TeX$  are first, then familiar familiar footnotes and finally the critical footnotes.

```
4946 \newcommand*{\l@ddoxtrafeet}{%
4947 \IfStrEq{familiar-critical}{\@fnpos}
4948 {\do@feetX\Xdo@feet}%
4949 {%
4950 \IfStrEq{critical-familiar}{\@fnpos}%
4951 {\Xdo@feet\do@feetX}%
4952 {\do@feetX\Xdo@feet}%
4953 }%
4954 }%
4955
4956 %
```

`\Xdo@feet` `\Xdo@feet` is the code extending `\@makecol` to cater for the extra critical feet.

```
4957 \newcommand*\Xdo@feet}{%
4958   \setbox\@outputbox \vbox{%
4959     \unvbox\@outputbox
4960     \@opXfeet}}
4961 %
```

`\@opXfeet` The extra critical feet to be added to the output. The normal way to add one series, `\print@Xnotes`, is replaced by `reledpar` when using `\Pages`.

```
4962 \newcommand\print@Xnotes[1]{%
4963   \xdef\@currentseries{#1}%
4964   \csuse{#1footstart}{#1}%
4965   \csuse{#1footgroup}{#1}%%
4966 }%
4967 %
```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```
4968 \newcommand*\@opXfeet}{%
4969   \unless\ifnocritical@%
4970     \gdef\firstXseries@{}%
4971     \def\do##1{%
4972       \ifvoid\csuse{##1footins}\else%
4973         \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4974         \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4975         \print@Xnotes{##1}%
4976         \fi%
4977       }%
4978     \dolistloop{\@series}%
4979     \fi%
4980 }%
4981 %
```

`\l@ddodoreintrafeet` `\l@ddodoreintrafeet` is the code for catering for the extra footnotes within `\@reinserts`. We use the same category and ordering as in `\l@ddoxtrafeet`.

```
4982 \newcommand*\l@ddodoreintrafeet}{%
4983   \IfStrEq{familiar-critical}{\@fnpos}
4984     {\@doreinfeetX\X@doreinfeet}%
4985     {%
4986     \IfStrEq{critical-familiar}{\@fnpos}%
4987       {\X@doreinfeet\@doreinfeetX}%
4988       {\@doreinfeetX\X@doreinfeet}%
4989     }%
4990 }
4991
4992 %
```

`\X@doreinfeet` `\X@doreinfeet` is the code for catering for the extra critical footnotes within `\@reinserts`.

```

4993 \newcommand*\X@doreinfeet}{%
4994   \unless\ifnocritical@%
4995   \def\do##1{%
4996     \ifvoid\csuse{##1footins}\else%
4997     \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4998     \fi}%
4999   \dolistloop{\@series}
5000   \fi%
5001 }
5002
5003 %

```

`\print@notesX` We have to add all the new kinds of familiar footnotes to the output routine. The normal way to add one series. `\print@Xnotes` is replaced by `reledpar` when using `\Pages`.

`\do@feetX`

`\@doreinfeetX`

```

5004 \newcommand\print@notesX[1]{%
5005   \xdef\@currentseries{#1}%
5006   \csuse{footstart#1}{#1}%
5007   \csuse{footgroup#1}{#1}%
5008 }%
5009 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

5010 \newcommand*\do@feetX{%
5011   \unless\ifnofamiliar@%
5012   \gdef\firstseriesX@{}%
5013   \setbox\@outputbox \vbox{%
5014     \unvbox\@outputbox%
5015     \def\do##1{%
5016       \ifvoid\csuse{footins##1}\else%
5017         \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
5018         \global\advance\skip\csuse{footins##1} by\csuse{afterruleX@##1}%
5019         \print@notesX{##1}%
5020       \fi%
5021     }%
5022     \dolistloop{\@series}}%
5023   \fi%
5024 }%
5025
5026 \newcommand{\@doreinfeetX}{%
5027   \unless\ifnofamiliar@%
5028   \def\do##1{%
5029     \ifvoid\csuse{footins##1}\else
5030     \insert%
5031     \csuse{footins##1}
5032     {\unvbox\csuse{footins##1}}%
5033     \fi%

```

```

5034 }%
5035 \dolistloop{\@series}%
5036 \fi%
5037 }%
5038
5039 %

```

### XXII.0.3 Standard output's commands patching

The memoir class does not use the ‘standard’ versions of `\@makecol` and `\@reinserts`, due to its sidebar insert. We had better add that code if memoir is used. (It can be awkward dealing with `\if` code within `\if` code, so don't use `\ifl@dmemoir` here.)

```

5040 \@ifclassloaded{memoir}{%
5041 %

```

memoir is loaded so we use memoir's built in hooks.

```

5042 \g@addto@macro{\m@doextrafeet}{\l@ddoxtrafeet}%
5043 \g@addto@macro{\m@ddodoreinextrafeet}{\l@ddodoreinextrafeet}%
5044 }{%
5045 %

```

memoir has not been loaded, so patch `\@makecol` and `\@reinserts`.

```

5046 \@ifpackageloaded{fancyhdr}{%
5047 \patchcmd%
5048   {\latex@makecol}%
5049   {\xdef\@freelist{\@freelist\@midlist}}%
5050   {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5051   {}%
5052   {\led@error@fail@patch@@@makecol}%
5053 }{%
5054 \patchcmd%
5055   {\@makecol}%
5056   {\xdef\@freelist{\@freelist\@midlist}}%
5057   {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5058   {}%
5059   {\led@error@fail@patch@@@makecol}%
5060 }%
5061
5062 \patchcmd%
5063   {\@reinserts}%
5064   {\ifvbox}%
5065   {\l@ddodoreinextrafeet\ifvbox}%
5066   {}%
5067   {\led@error@fail@patch@@@reinserts}%
5068 }
5069
5070 %

```

It turns out that `\doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```
5071 \newif\if@led@nofoot
5072
5073 %

5074 \@ifclassloaded{memoir}{%
5075 %
```

If the `memoir` class is loaded we hook into its modified `\doclearpage`.

```
\@mem@extranofeet 76 \g@addto@macro{\@mem@extranofeet}{%%
5077   \def\do#1{%
5078     \unless\ifnocritical@%
5079       \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
5080     \fi%
5081     \unless\ifnofamiliar@%
5082       \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
5083     \fi%
5084   }
5085   \dolistloop{\@series}%
5086 }%
5087 }{%
5088 %
```

As `memoir` is not loaded we have patch `\doclearpage`.

```
\@led@testifnofoot 89 \newcommand*\@led@testifnofoot}{%
\@doclearpage 90   \@led@nofoottrue%
5091   \ifvoid\footins\else%
5092     \@led@nofootfalse%
5093   \fi%
5094   \def\do##1{%
5095     \unless\ifnocritical@%
5096       \ifvoid\csuse{##1footins}\else%
5097         \@led@nofootfalse%
5098       \fi%
5099     \fi%
5100     \unless\ifnofamiliar@%
5101       \ifvoid\csuse{footins##1}\else%
5102         \@led@nofootfalse%
5103       \fi%
5104     \fi%
5105   }%
5106   \dolistloop{\@series}%
5107 }%
5108
```

```

5109 \pretocmd%
5110   {\@docclearpage}%
5111   {\@led@testifnofoot}%
5112   {}%
5113   {\led@error@fail@patch@@docclearpage}%
5114
5115 \patchcmd%
5116   {\@docclearpage}%
5117   {\ifvoid\footins}%
5118   {\if@led@nofoot}%
5119   {}%
5120   {\led@error@fail@patch@@docclearpage}%
5121
5122 }
5123
5124 %

```

## XXIII Cross referencing

You can mark a place in the text using a command of the form `\edlabel{<foo>}`, and later refer to it using the label `<foo>` by typing `\edpageref{<foo>}`, or `\lineref{<foo>}` or `\sublineref{<foo>}` or `\pstartref`. These reference commands will produce, respectively, the page, line sub-line and pstart on which the `\edlabel{<foo>}` command occurred.

The reference macros warn you if a reference is made to an undefined label. If `{<foo>}` has been used as a label before, the `\edlabel{<foo>}` command will issue a complaint; subsequent `\edpageref` and `\edlineref` commands will refer to the latest occurrence of `\edlabel{<foo>}`.

`\labelref@list` Set up a new list, `\labelref@list`, to hold the page, line and sub-line numbers for each label.

```

5125 \list@create{\labelref@list}
5126 %

```

`\zz@@@` A convenience macro to zero two labeling counters in one go.

```

5127 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
5128
5129 %

```

`\edlabel` The `\edlabel` command first writes a `\@lab` macro to the `\linenum@out` file. It then checks to see that the `\labelref@list` actually has something in it (if not, it creates a dummy entry), and pops the next value for the current label, storing it in `\label@refs`. Finally it defines the label to be `\empty` so that any future check will turn up the fact that it has been used.<sup>31</sup>

<sup>31</sup>The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

This version of the original `edmac \label` uses `\@bsphack` and `\@esphack` to eliminate extra space problems and also use the  $\LaTeX$  write methods for the `.aux` file.

Jesse Billett<sup>32</sup> found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```

5130 \newcommand*\edlabel}[1]{%
5131   \ifl@dpairing\ifautopar%
5132     \strut%
5133     \fi\fi%
5134     \@bsphack%
5135     \ifboolexpr{bool{ledRcol} or bool{ledRcol@}}{%
5136       \ifXnote@%
5137         \protected@write\@auxout{%
5138           {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub|\the\c@pstartR|{#1}}%
5139         \ifdef{\hypertarget}%
5140           {\Hy@raisedlink{\hypertarget{#1}{}}}%
5141           {}%
5142         \else%
5143           \write\linenum@outR{\string\@lab}%
5144           \ifx\labelref@listR\empty%
5145             \xdef\label@refs{\zz@@@}%
5146           \else%
5147             \gl@p\labelref@listR\to\label@refs%
5148           \fi%
5149           \ifvmode%
5150             \advancelabel@refs%
5151           \fi%
5152 %

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

5153   \protected@write\@auxout{%
5154     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
|{#1}}%
5155   \ifdef{\hypertarget}%
5156     {\Hy@raisedlink{\hypertarget{#1}{}}}%
5157     {}%
5158   \fi%
5159 }{%
5160   \ifXnote@%
5161     \protected@write\@auxout{%
5162       {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\
l@dparsedstartsub|\the\c@pstartR|{#1}}%
5163     \ifdef{\hypertarget}%
5164       {\Hy@raisedlink{\hypertarget{#1}{}}}%
5165       {}%
5166     \else%

```

<sup>32</sup>(jdb43@cam.ac.uk) via the ctt thread 'ledmac cross referencing', 25 August 2003.



```

5167 \write\linenum@out{\string\@lab}%
5168 \ifx\labelref@list\empty%
5169 \xdef\label@refs{\zz@@}%
5170 \else%
5171 \gl@p\labelref@list\to\label@refs%
5172 \fi%
5173 \ifvmode%
5174 \advance\label@refs%
5175 \fi%
5176 \protected@write\@auxout{%
5177 {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstart
|{#1}}%
5178 \ifdef{\hypertarget}%
5179 {\Hy@raisedlink{\hypertarget{#1}{}}}%
5180 {}%
5181 \fi%
5182 }%
5183 \@esphack}%
5184
5185 %

```

`\advance\label@refs` In cases where `\edlabel` is the first element in a paragraph, we have a problem with line counts, because line counts change only at the first horizontal box of the paragraph. Hence, we need to test `\edlabel` if it occurs at the start of a paragraph. To do so, we use `\ifvmode`. If the test is true, we must advance by one unit the amount of text we write into the `.aux` file. We do so using `\advance\label@refs` command.

```

5186 \newcounter{line}%
5187 \newcounter{subline}%
5188 \newcommand{\advance\label@refs}{%
5189 \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
5190 \stepcounter{line}%
5191 \ifsublines@%
5192 \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
5193 }%
5194 \stepcounter{subline}{1}%
5195 \def\label@refs{\theline|\thesubline}%
5196 \else%
5197 \def\label@refs{\theline|0}%
5198 \fi%
5199 }
5200 \def\labelrefsparseline#1|#2{#1}
5201 \def\labelrefsparsesubline#1|#2{#2}
5202 %

```

`\l@dmake@labels` The `\l@dmake@labels` macro gets executed when the labels file is read. For each label it defines a macro, whose name is made up partly from the label you supplied, that contains the page, line and sub-line numbers. But first it checks to see whether the label has already been used (and complains if it has).

The initial use of `\newcommand` is to catch if `\l@dmake@labels` has been previously defined (by a class or package).

#1 page number, #2 line number, #3 sub-line number, #4 pstart number, #5 label.

```

5202 \newcommand*\l@dmake@labels}{
5203 \def\l@dmake@labels#1|#2|#3|#4|#5{%
5204 \expandafter\ifx\csname the@label\csuse{XR@prefix}#5\endcsname \relax\
else
5205 \led@warn@DuplicateLabel{\csuse{XR@prefix}#5}%
5206 \fi
5207 \expandafter\gdef\csname the@label\csuse{XR@prefix}#5\endcsname
{#1|#2|#3|#4|\relax}%
5208 \ignorespaces}
5209
5210 %

```

$\TeX$  reads the aux file at both the beginning and end of the document, so we have to switch off duplicate label checking after the first time the file is read.

```

5211 \AtBeginDocument{%
5212 \def\l@dmake@labels#1|#2|#3|#4|#5{%
5213 }
5214
5215 %

```

`\@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.

$\TeX$  uses the page counter for page numbers. However, it appears that this is not the right place to grab the page number. That task is now done in the `\edlabel` macro. This version of `\@lab` appends just the current line and sub-line numbers to `\labelref@list`.

```

5216
5217 \newcommand*\@lab}{%
5218 \ifledRcol
5219 \xright@appenditem{\linenumr@p{\line@numR}}|%
5220 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
5221 \to\labelref@listR
5222 \else
5223 \xright@appenditem{\linenumr@p{\line@num}}|%
5224 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
5225 \to\labelref@list
5226 \fi}
5227 %

```

`\applabel` `\applabel`, if called in `\edtext` will insert automatically both a start and an end label for the current edtext lines.

```

5228 \newcommand*{\applabel}[1]{%
5229   \ifnum\@edtext@level>0%
5230   %

```

Label should not be already defined.

```

5231   \ifcsundef{the@label#1}{%
5232     \csdef{the@label#1}{applabel}%
5233   }%
5234   {%
5235     \led@warn@DuplicateLabel{#1 (applabel)}%
5236   }%
5237 %

```

Parse the `\edtext` line numbers.

```

5238   \expandafter\l@dp@rsefootspec\l@d@nums|%
5239 %

```

Use the  $\TeX$  standard hack for label.

```

5240   \@bspack%
5241 %

```

And now, write the data in the auxiliary file.

```

5242   \ifledRcol%
5243     \protected@write\@auxout{}%
5244       {\string\l@dmake@labelsR\space\l@dparsedstartpage|\
5245 \l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
5246     \ifdef{\hypertarget}%
5247       {\Hy@raisedlink{\hypertarget{#1:start}}}%
5248     {}%
5249     \protected@write\@auxout{}%
5250       {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline
5251 |\l@dparsedendsub|\the\c@pstartR|{#1:end}}%
5252   \else%
5253     \protected@write\@auxout{}%
5254       {\string\l@dmake@labels\space\l@dparsedstartpage|\
5255 \l@dparsedstartline|\l@dparsedstartsub|\the\c@pstart|{#1:start}}%
5256     \ifdef{\hypertarget}%
5257       {\Hy@raisedlink{\hypertarget{#1:start}}}%
5258     {}%
5259     \protected@write\@auxout{}%
5260       {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline
5261 |\l@dparsedendsub|\the\c@pstart|{#1:end}}%
5262   \fi%
5263 %

```

Use the  $\TeX$  standard hack for label.

```

5260   \@espack%
5261 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

5262 \else%
5263 \led@warn@AppLabelOutEdtext{#1}%
5264 \fi%
5265 %

End of \applabel

5266 }%
5267 %

```

`\edlabelS` `\edlabelS` and `\edlabelE` are just used to mark the beginning and the end of a passage.

`\edlabelE`

`\edlabelSE`

```

5268 \newcommand{\edlabelS}[1]{%
5269 \edlabel{#1:start}%
5270 }
5271 \newcommand{\edlabelE}[1]{%
5272 \edlabel{#1:end}%
5273 }
5274 \newcommand{\edlabelSE}[1]{%
5275 \edlabelS{#1}%
5276 \edlabelE{#1}%
5277 }
5278 %

```

`\wrap@edcrossref` `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

5279 \newrobustcmd{\wrap@edcrossref}[2]{%
5280 \ifdef{hyperlink}%
5281 {\hyperlink{#1}{#2}}%
5282 {#2}%
5283 }
5284 %

```

`\edpageref` If the specified label exists, `\edpageref` gives its page number.

`\xpageref` For this reference command, as for the other two, a special version with prefix `x` is provided for use in places where the command is to be scanned as a number, as in `\linenum`. These special versions have two limitations: they do not print error messages if the reference is unknown, and they can't appear as the first label or reference command in the file; you must ensure that a `\edlabel` or a normal reference command appears first, or these `x`-commands will always return zeros.

TeX already defines a `\pageref`, so changing the name to `\edpageref`.

```

5285 \newcommand*{\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}
5286 \newcommand*{\xpageref}[1]{\l@dgetref@num{1}{#1}}
5287
5288 %

```

`\edlineref` If the specified label exists, `\lineref` gives its line number.

`\xlineref`

```

5289 \newcommand*\edlineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\
l@dgetref@num{2}{#1}\xflagref{#1}}}%
5290 \newcommand*\xlineref}[1]{\l@dgetref@num{2}{#1}}%
5291
5292 %

```

**\sublineref** If the specified label exists, `\sublineref` gives its sub-line number.

**\xsublineref**

```

5293 \newcommand*\sublineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\
l@dgetref@num{3}{#1}}
5294 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}
5295
5296 %

```

**\pstarteref** If the specified label exists, `\pstarteref` gives its pstart number.

**\xpstarteref**

```

5297 \newcommand*\pstarteref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\
l@dgetref@num{4}{#1}}
5298 \newcommand*\xpstarteref}[1]{\l@dgetref@num{4}{#1}}
5299
5300 %

```

**\xflagref** `\xflagref` finds the side flag of any ref defined with `\edlabel`.

```

5301 \newcommand*\xflagref}[1]{\l@dgetref@num{5}{#1}}
5302 %

```

The next three macros are used by the referencing commands above, and do the job of extracting the right numbers from the label macro that contains the page, line, and sub-line number.

**\l@dref@undefined** The `\l@dref@undefined` macro is called when you refer to a label with the normal referencing macros. Its argument is a label, and it just checks that the label has been defined.

```

5303 \newcommand*\l@dref@undefined}[1]{%
5304   \expandafter\ifx\csname the@label#1\endcsname\relax
5305     \led@warn@RefUndefined{#1}%
5306   \fi}
5307
5308 %

```

**\l@dgetref@num** Next, `\l@dgetref@num` fetches the number we want. It has two arguments: the first is simply a digit, specifying whether to fetch a page (1), line (2), sub-line (3), (4) pstart number or (5) side flag. (This switching is done by calling `\l@dlabel@parse`.) The second argument is the label-macro, which because of the `\@lab` macro above is defined to be a string of the type 123|456|789.

```

5309 \newcommand*\l@dgetref@num}[2]{%
5310   \expandafter
5311   \ifx\csname the@label#2\endcsname \relax
5312     000%
5313   \else
5314     \expandafter\expandafter\expandafter
5315     \l@dlabel@parse\csname the@label#2\endcsname|#1%
5316   \fi}
5317
5318 %

```

`\l@dlabel@parse` Notice that we slipped another | delimiter into the penultimate line of `\l@dgetref@num`, to keep the ‘switch-number’ separate from the reference numbers. This | is used as another parameter delimiter by `\l@dlabel@parse`, which extracts the appropriate number from its first arguments. The |-delimited arguments consist of the expanded label-macro (three reference numbers), followed by the switch-number (1, 2, 3 or 4) which defines which of the earlier five numbers to pick out. (It was earlier given as the first argument of `\l@dgetref@num`.)

```

5319 \newcommand*\l@dlabel@parse}{%
5320 \def\l@dlabel@parse#1|#2|#3|#4|#5|#6{%
5321   \ifcase #6%
5322     \or #1%
5323     \or #2%
5324     \or #3%
5325     \or #4%
5326     \or #5%
5327   \fi}
5328 %

```

`\xxref` The `\xxref` command takes two arguments, both of which are labels, e.g., `\xxref{mouse}{elephant}`. It first does some checking to make sure that the labels do exist (if one does not, those numbers are set to zero). Then it calls `\linenum` and sets the beginning page, line, and sub-line numbers to those of the place where `\label{mouse}` was placed, and the ending numbers to those at `{elephant}`. The point of this is to be able to manufacture footnote line references to passages which cannot be specified in the normal way as the first argument to `\edtext` for one reason or another. Using `\xxref` in the second argument of `\edtext` lets you set things up at least semi-automatically.

```

5329 \newcommand*\xxref}[2]{%
5330   {%
5331     \expandafter\ifx\csname the@label#1\endcsname \relax%
5332       \expandafter\let\csname the@@label#1\endcsname\zz@@%
5333     \else%
5334       \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
5335       {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
5336       \fi%
5337     \expandafter\ifx\csname the@label#2\endcsname \relax%
5338       \expandafter\let\csname the@@label#2\endcsname\zz@@%

```

```

5338 \else%
5339 \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
{1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
5340 \fi%
5341 \letcs{\@tempa}{the@@label#1}%
5342 \letcs{\@tempb}{the@@label#2}%
5343 \linenum{\@tempa|%
5344 \@tempb}}}%
5345
5346 %

```

`\appref` `\appref`, `\Seref`, `\apprefwithpage`, `\Serefwithpage` and `\SEonlypage` print cross-ref to some start / end lines defined by specific commands. It prints the lines as they should be printed in the apparatus (critical notes for not suffixed versions, endnotes for suffixed versions).

`\Serefwithpage` Here we define hooks similar to some those related to critical footnotes or endnotes. So, first declare the default value of the hooks for the pseudo-series. Also declare the internal toggle which are switch by `reledmac`.

```

5347 \def\Xtwolines@appref{}%
5348 \def\Xtwolines@Seref{}%
5349
5350 \def\Xmorethantwolines@appref{}%
5351 \def\Xmorethantwolines@Seref{}%
5352
5353 \def\Xlinerangeseparator@appref{\endashchar}%
5354 \def\Xlinerangeseparator@Seref{\endashchar}%
5355
5356 \def\Xsublinesep@appref{\fullstop}%
5357 \def\Xsublinesep@Seref{\fullstop}%
5358
5359 \newtoggle{Xtwolinesbutnotmore@appref}%
5360 \newtoggle{Xtwolinesbutnotmore@Seref}%
5361
5362 \newtoggle{Xtwolinesonlyinsamepage@appref}%
5363
5364 \newtoggle{Xtwolinesonlyinsamepage@Seref}%
5365
5366 \newtoggle{Xlineflag@appref}%
5367 \toggletrue{Xlineflag@appref}%Here exception
5368 \newtoggle{Xlineflag@Seref}%
5369 \toggletrue{Xlineflag@Seref}%%Here exception
5370
5371 \def\Xendtwolines@apprefwithpage{}%
5372 \def\Xendtwolines@Serefwithpage{}%
5373
5374 \def\Xendmorethantwolines@apprefwithpage{}%
5375 \def\Xendmorethantwolines@Serefwithpage{}%
5376

```

```

5377 \def\Xendlinerrangeseparator@apprefwithpage{\endashchar}
5378 \def\Xendlinerrangeseparator@SErefwithpage{\endashchar}
5379 \def\Xendlinerrangeseparator@SErefonlypage{\endashchar}
5380
5381 \def\Xendbeforepagenumber@apprefwithpage{p.}%
5382 \def\Xendbeforepagenumber@SErefwithpage{p.}%
5383 \def\Xendbeforepagenumber@SEonlypage{p.}%
5384
5385 \def\Xendafterpagenumber@apprefwithpage{ }%
5386 \def\Xendafterpagenumber@SErefwithpage{ }%
5387
5388
5389 \def\Xendlineprefixsingle@apprefwithpage{ }%
5390 \def\Xendlineprefixsingle@SErefwithpage{ }%
5391
5392 \def\Xendlineprefixmore@apprefwithpage{ }%
5393 \def\Xendlineprefixmore@SErefwithpage{ }%
5394
5395 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
5396 \newtoggle{Xendtwolinesbutnotmore@SErefwithpage}%
5397
5398 \def\Xendsublinesep@apprefwithpage{\fullstop}%
5399 \def\Xendsublinesep@SErefwithpage{\fullstop}%
5400
5401 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
5402 \newtoggle{Xendtwolinesonlyinsamepage@SErefwithpage}%
5403
5404 \newtoggle{Xendlineflag@apprefwithpage}
5405 \toggletrue{Xendlineflag@apprefwithpage}%Here, exception
5406 \newtoggle{Xendlineflag@SErefwithpage}
5407 \toggletrue{Xendlineflag@SErefwithpage}%Here, exception
5408
5409 %

```

Note that some of these hooks are declared but no user command can change their values. Such hooks are not pertinent for appref and apprefwithpage pseudo-series, but their values are nonetheless tested in some macros.

```

5410
5411 \xdef\Xboxstartlinenum@appref{Opt}
5412 \xdef\Xboxstartlinenum@SEref{Opt}
5413
5414 \xdef\Xboxendlinenum@appref{Opt}
5415 \xdef\Xboxendlinenum@SEref{Opt}
5416
5417 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
5418 \xdef\Xendboxstartlinenum@SErefwithpage{Opt}
5419
5420 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
5421 \xdef\Xendboxendlinenum@SErefwithpage{Opt}
5422

```



```

5423 %
    Now, declare the default values of \@apprefprefixsingle and \@apprefprefixmore,
    \@SErefprefix, \@SErefprefixmore and the commands which defines them.
5424 \newcommand\@apprefprefixsingle{}%
5425 \newcommand\@SErefprefixsingle{}%
5426
5427 \newcommand\@apprefprefixmore{}%
5428 \newcommand\@SErefprefixmore{}%
5429
5430 \newcommand{\setapprefprefixsingle}[1]{%
5431   \gdef\@apprefprefixsingle{#1}%
5432 }
5433 \newcommand{\setSErefprefixsingle}[1]{%
5434   \gdef\@SErefprefixsingle{#1}%
5435 }
5436
5437 \newcommand{\setapprefprefixmore}[1]{%
5438   \gdef\@apprefprefixmore{#1}%
5439 }
5440 \newcommand{\setSErefprefixmore}[1]{%
5441   \gdef\@SErefprefixmore{#1}%
5442 }
5443
5444 %

```

And not `\setSErefonlypageprefixsingle` and `\setSErefonlypageprefixmore`.

```

5445 \let\setSErefonlypageprefixsingle\XendbeforepagenumberSErefonlypage%
5446 \newcommand{\setSErefonlypageprefixmore}[1]{%
5447   \gdef\SErefonlypage@prefixmore{#1}%
5448 }%
5449 %

```

And now, the main commands: `\appref`, `\apprefwithpage`, `\SEref` and `\SErefwithpage`. These commands call `\reformatted@` and `\reformattedwithpage`, which calls `\printlines` and `\printendlines`. That is why we have previously declared all hooks values tested inside these last commands.

```

5450
5451 \newcommandx{\appref}[2][1,usedefault]{\reformatted@{#1}{#2}{appref}}
5452 \newcommandx{\SEref}[2][1,usedefault]{\reformatted@{#1}{#2}{SEref}}
5453
5454 \newcommandx{\apprefwithpage}[2][1,usedefault]{\reformattedwithpage@
5455 {#1}{#2}{appref}}
5456 \newcommandx{\SErefwithpage}[2][1,usedefault]{\reformattedwithpage@
5457 {#1}{#2}{SEref}}
5458 \newcommandx{\SErefonlypage}[2][1,usedefault]{\reformattedonlypage@
5459 {#1}{#2}{SEref}}

```

```

5459 \newcommand{\reformatted@}[3]{%
5460   \def\do##1{%
5461     \setkeys[mac]{truefootnoteoption}{##1}%
5462   }%
5463   \notblank{#1}{\docsvlist{#1}}{-%
5464   \xdef\@currentseries{#3}%
5465   \ifcempty{@#3prefixmore}%
5466     {\@apprefprefixsingle}%
5467     {%
5468       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
5469       {\csuse{@#3prefixsingle}}%
5470       {\csuse{@#3prefixmore}}%
5471     }%
5472   \ifboolexpr{%
5473     test{\ifcsundef{the@label#2:start}}%
5474     or test{\ifcsundef{the@label#2:end}}%
5475   }%
5476   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
5477   {%
5478     \def\@this@crossref@start{#2:start}%
5479     \def\@this@crossref@end{#2:end}%
5480     \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|}%
5481     \undef\@this@crossref@end%
5482     \undef\@this@crossref@start%
5483   }%
5484   \def\do##1{%
5485     \setkeys[mac]{falsefootnoteoption}{##1}%
5486   }%
5487   \notblank{#1}{\docsvlist{#1}}{-%
5488   }%
5489
5490 \newcommand{\reformattedwithpage@}[3]{%
5491   \def\do##1{%
5492     \setkeys[mac]{truefootnoteoption}{##1}%
5493   }%
5494   \notblank{#1}{\docsvlist{#1}}{-%
5495   \xdef\@currentseries{#3withpage}%
5496   \ifboolexpr{%
5497     test{\ifcsundef{the@label#2:start}}%
5498     or test{\ifcsundef{the@label#2:end}}%
5499   }%
5500   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
5501   {%
5502     \def\@this@crossref@start{#2:start}%
5503     \def\@this@crossref@end{#2:end}%
5504     \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|}%

```

```

5505 \undef\@this@crossref@end%
5506 \undef\@this@crossref@start%
5507 }%
5508 \def\do##1{%
5509 \setkeys[mac]{falsefootnoteoption}{##1}%
5510 }%
5511 \notblank{#1}{\docsvlist{#1}}{-}%
5512 }%
5513
5514 \newcommand{\reformattedonlypage@}[3]{%
5515 \def\do##1{%
5516 \setkeys[mac]{truefootnoteoption}{##1}%
5517 }%
5518 \notblank{#1}{\docsvlist{#1}}{-}%
5519 \xdef\@currentseries{#3onlypage}%
5520 \ifboolexpr{%
5521 test{\ifcsundef{the@label#2:start}}%
5522 or test{\ifcsundef{the@label#2:end}}%
5523 }%
5524 {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
5525 {\ifnumequal{\xpageref{#2:end}}{\xpageref{#2:start}}%
5526 {%
5527 \printnpnum{%
5528 \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
5529 }%
5530 }%
5531 {%
5532 \ifcsvoid{#3onlypage@prefixmore}%
5533 {}%
5534 {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixmore}}%
5535 \ifdefined\linerangesep@%
5536 \printnpnum{%
5537 \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
5538 \linerangesep@%
5539 \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
5540 }%
5541 \else%
5542 \printnpnum{%
5543 \wrap@edcrossref{#2:start}{\xpageref{#2:start}}%
5544 \csuse{Xendlinerangeseparator@\@currentseries}%
5545 \wrap@edcrossref{#2:end}{\xpageref{#2:end}}%
5546 }%
5547 \fi%
5548 }%
5549 }%
5550 \def\do##1{%
5551 \setkeys[mac]{falsefootnoteoption}{##1}%
5552 }%
5553 \notblank{#1}{\docsvlist{#1}}{-}%
5554 }%

```

```
5555 %
```

`\edmakeLabel` Sometimes the `\edlabel` command cannot be used to specify exactly the page and line desired; you can use the `\edmakeLabel` macro make your own label. For example, if you insert `\edmakeLabel{elephant}{10|25|0}` you will have created a new label, and a later call to `\edpageref{elephant}` would print ‘10’ and `\lineref{elephant}` would print ‘25’. The sub-line number here is zero. `\edmakeLabel` takes a label, followed by a page and a line number(s) as arguments.  $\TeX$  defines a `\makeLabel` macro which is used in lists. Peter Wilson has changed the name to `\edmakeLabel`.

```
5556 \newcommand*\edmakeLabel[2]{\expandafter\xdef\csname the@label#1\
endcsname{#2}}
```

```
5557
```

```
5558 %
```

(If you are only going to refer to such a label using `\xxref`, then you can omit entries in the same way as with `\linenum` (see VI.3 p. 117 and V.9 p. 87), since `\xxref` makes a call to `\linenum` in order to do its work.)

### XXIII.1 Compatibility with xref

Here, we provide compatibility with the `xref` to enable `reledmac`’s cross-referencing to external documents. We assume that the user loads `xref` *before* `reledmac`, but uses `\externaldocument` *after* loading `reledmac`.

`\XR@test` First, we patch the `xr` macro `\XR@test`, which is called on every line of the external `.aux` file, in order to also call macros specific to `reledmac`.

```
5559 \pretocmd{\XR@test}%
5560   {\XR@test@mac+++#1#2#3#4+++}%
5561   {}%
5562   {}%
5563 %
```

`\XR@test@mac` The `\XR@test@mac` takes the full content of a line of the external `.aux` files, with the three final dots added by `xr`.

```
5564 \long\def\xR@test@mac+++#1+++{\XR@test@mac@test#1}
5565 %
```

`\XR@test@mac@test` And finally, `\XR@test@mac@test` does the job. This code is based on the `\XR@test` macro of the `xr` package. However, note that the `\XR@` prefix is not called here, but it is integrated directly in `\l@dmake@labels` and `\l@dmake@labelsR`.

```
5566 \long\def\xR@test@mac@test#1#2...{%The triple dots (NOT \ldots) are because
of the line 22 of xr.sty v5.02 1994/05/28
5567   \ifx#1\l@dmake@labels%
5568     \l@dmake@labels#2%
5569   \else
```

```

5570 \ifx#1\l@dmake@labelsR%
5571 \l@dmake@labelsR #2%
5572 \fi%
5573 \fi%
5574 }%
5575 %

```

## XXIV Side notes

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

`\@xympar` Changing `\@xympar` a little at least ensures that `\marginpar` in numbered text do not disturb the flow.

```

5576 \pretocmd{\@xympar}%
5577 {\ifnumberedpar@
5578 \led@warn@NoMarginpars
5579 \@esphack
5580 \else}%
5581 {}%
5582 {}%
5583
5584 \apptocmd{\@xympar}%
5585 {\fi}%
5586 {}
5587 {}
5588
5589 %

```

We provide side notes as replacement for `\marginpar` in numbered text.

`\sidenote@margin` These are the sidenote equivalents to `\line@margin` and `\linenummargin` for specifying which margin. The default is the right margin (opposite to the default for line numbers). `\l@dgetsidenote@margin` returns the number associated to side note margin:

**left** : 0

**right** : 1

**outer** : 2

**inner** : 3

```

5590 \newcount\sidenote@margin
5591 \newcommand*{\sidenotemargin}[1]{\{
5592 \l@dgetsidenote@margin{#1}%
5593 \ifnum\l@dttempcntb>\m@ne

```

```

5594 \ifledRcol
5595 \global\sidenote@marginR=\@l@tempcntb
5596 \else
5597 \global\sidenote@margin=\@l@tempcntb
5598 \fi
5599 \fi}}
5600 \newcommand*\l@dgetsidenote@margin}[1]{%
5601 \def\@tempa{#1}\def\@tempb{left}%
5602 \ifx\@tempa\@tempb
5603 \l@tempcntb \z@
5604 \else
5605 \def\@tempb{right}%
5606 \ifx\@tempa\@tempb
5607 \l@tempcntb \@ne
5608 \else
5609 \def\@tempb{outer}%
5610 \ifx\@tempa\@tempb
5611 \l@tempcntb \tw@
5612 \else
5613 \def\@tempb{inner}%
5614 \ifx\@tempa\@tempb
5615 \l@tempcntb \thr@@
5616 \else
5617 \led@warn@BadSidenotemargin
5618 \l@tempcntb \m@ne
5619 \fi
5620 \fi
5621 \fi
5622 \fi}
5623 \sidenotemargin{right}
5624
5625 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
5626 \newbox\l@dlp@rbox
5627 \newbox\l@drp@rbox
5628
5629 %

```

`\ledlsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`), their  
`\ledrsnotewidth` distance from the text (initialised to `\linenumsep`), and the fonts used.

```

\ledlsnotesep
5630 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep
5631 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup
5632 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup
5633 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
5634 \newcommand*\ledlsnotefontsetup}{\raggedleft\footnotesize}
5635 \newcommand*\ledrsnotefontsetup}{\raggedright\footnotesize}
5636

```

```

5637 %
\ledleftnote \ledleftnote, \ledrightnote, \ledinnernote, \ledouternote are the user com-
\ledrightnote mands for left, right, inner and outer sidenotes. The two last one are just alias for the
\ledinnernote two first one, depending of the page number. \ledsidenote{<text>} is the command
\ledouterote for a moveable sidenote.
\ledsidenote
5638 \newcommand*{\ledleftnote}[1]{\edtext{}{\l@dlsnote{#1}}}
5639 \newcommand*{\ledrightnote}[1]{\edtext{}{\l@drsnote{#1}}}
5640
5641 \newcommand*{\ledinnernote}[1]{%
5642 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5643 \ledleftnote{#1}%
5644 \else%
5645 \ledrightnote{#1}%
5646 \fi%
5647 }
5648
5649 \newcommand*{\ledouternote}[1]{%
5650 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5651 \ledrightnote{#1}%
5652 \else%
5653 \ledleftnote{#1}%
5654 \fi%
5655 }
5656
5657 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcsnote{#1}}}
5658 %

```

**\l@dlsnote** . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

**\l@drsnote**

**\l@dcsnote**

```

5659 \newif\ifrightnoteup
5660 \rightnoteuptrue
5661
5662 \newcommand*{\l@dlsnote}[1]{%
5663 \begingroup%
5664 \newcommand{\content}{#1}%
5665 \ifnumberedpar@
5666 \ifledRcol%
5667 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}%
5668 \to\inserts@listR
5669 \global\advance\insert@countR \@ne%
5670 \else%
5671 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}%
5672 \to\inserts@list
5673 \global\advance\insert@count \@ne%
5674 \fi

```

```

5675 \fi\ignorespaces\endgroup}
5676
5677 \newcommand*{\l@drsnote}[1]{%
5678 \begingroup%
5679 \newcommand{\content}{#1}%
5680 \ifnumberedpar@
5681 \ifledRcol%
5682 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}%
5683 \to\inserts@listR
5684 \global\advance\insert@countR \@ne%
5685 \else%
5686 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}%
5687 \to\inserts@list
5688 \global\advance\insert@count \@ne%
5689 \fi
5690 \fi\ignorespaces\endgroup}
5691
5692 \newcommand*{\l@dcsnote}[1]{%
5693 \begingroup%
5694 \newcommand{\content}{#1}%
5695 \ifnumberedpar@
5696 \ifledRcol%
5697 \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}%
5698 \to\inserts@listR
5699 \global\advance\insert@countR \@ne%
5700 \else%
5701 \xright@appenditem{\noexpand\vl@dcsnote{\expandonce\content}}%
5702 \to\inserts@list
5703 \global\advance\insert@count \@ne%
5704 \fi
5705 \fi\ignorespaces\endgroup}
5706
5707 %

```

`\vl@dlsnote` Put the left/right text into boxes, but just save the moveable text. `\l@dcsnotetext`, `\vl@drsnote` `\l@dcsnotetext@l` and `\l@dcsnotetext@r` are `etoolbox`'s lists which will store the content of side notes. We store the content in lists, because we need to loop later on them, in case many sidenote co-exist for the same line. That is there some special test to do, in order to:

- Store the content of `\ledsidenote` to `\l@dcsnotetext` in any cases.
- Store the content of `\rightsidenote` to:
  - `\l@dcsnotetext` if `\ledsidenote` is to be put on right.
  - `\l@dcsnotetext@r` if `\ledsidenote` is to be put on left.
- Store the content of `\leftsidenote` to:
  - `\l@dcsnotetext` if `\ledsidenote` is to be put on left.



- \l@dcsnotetext@l if \ledsidenote is to be put on right.

```

5708 \newcommand*{\vl@dlsnote}[1]{%
5709   \ifledRcol@%
5710     \@l@dttempcntb=\sidenote@marginR%
5711     \ifnum\@l@dttempcntb>\@ne%
5712       \advance\@l@dttempcntb by\page@numR%
5713     \fi%
5714   \else%
5715     \@l@dttempcntb=\sidenote@margin%
5716     \ifnum\@l@dttempcntb>\@ne%
5717       \advance\@l@dttempcntb by\page@num%
5718     \fi%
5719   \fi%
5720   \ifodd\@l@dttempcntb%
5721     \listgadd{\l@dcsnotetext@l}{#1}%
5722   \else%
5723     \listgadd{\l@dcsnotetext}{#1}%
5724   \fi
5725 }
5726 \newcommand*{\vl@drsnote}[1]{%
5727   \ifledRcol@%
5728     \@l@dttempcntb=\sidenote@marginR%
5729     \ifnum\@l@dttempcntb>\@ne%
5730       \advance\@l@dttempcntb by\page@numR%
5731     \fi%
5732   \else%
5733     \@l@dttempcntb=\sidenote@margin%
5734     \ifnum\@l@dttempcntb>\@ne%
5735       \advance\@l@dttempcntb by\page@num%
5736     \fi%
5737   \fi%
5738   \ifodd\@l@dttempcntb%
5739     \listgadd{\l@dcsnotetext}{#1}%
5740   \else%
5741     \listgadd{\l@dcsnotetext@r}{#1}%
5742   \fi%
5743 }
5744 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcsnotetext}{#1}}
5745
5746 %

```

`\setl@dlp@rbox` `\setl@dlprbox{<lednums>}{<tag>}{<text>}` puts `<text>` into the `\l@dlp@rbox` box.  
`\setl@drpr@box` And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```

5747 \newcommand*{\setl@dlp@rbox}[1]{%
5748   {\parindent\z@\hspace=\ledlsnotewidth\ledlsnotefontsetup
5749     \global\setbox\l@dlp@rbox
5750     \ifleftnoteup

```

```

5751     =\vbox to\z@\vss #1}%
5752     \else
5753     =\vbox to 0.70\baselineskip{\strut#1\vss}%
5754     \fi}}
5755 \newcommand*\setl@drp@rbox}[1]{%
5756   {\parindent\z@\hsize=\ledrsnotewidth\ledrsnotefontsetup
5757     \global\setbox\l@drp@rbox
5758     \ifrightnoteup
5759     =\vbox to\z@\vss#1}%
5760     \else
5761     =\vbox to0.7\baselineskip{\strut#1\vss}%
5762     \fi}}
5763 \newif\ifleftnoteup
5764 \leftnoteuptrue
5765 %

```

`\@sidenoteseq` This macro is used to separate sidenotes of the same line.

```

5766 \newcommand{\setsidenoteseq}[1]{\gdef\@sidenoteseq{#1}}
5767 \newcommand{\@sidenoteseq}{, }
5768 %

```

`\affixside@note` This macro puts any moveable sidenote text into the left or right sidenote box, depending on which margin it is meant to go in. It's a very much stripped down version of `\affixlin@num`.

Before do it, we concatenate all moveable sidenotes of the line, using `\@sidenoteseq` as separator. It is the result that we put on the sidenote.

```

5769 \newcommand*\affixside@note}{%
5770   \def\sidenotecontent@{}%
5771   \numgdef\itemcount@{0}%
5772   \def\do##1{%
5773     \ifnumequal{\itemcount@}{0}%
5774     {%
5775       \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
5776     {\appto\sidenotecontent@{\@sidenoteseq ##1}%
5777     }%
5778     \numgdef\itemcount@{\itemcount@+1}%
5779   }%
5780   \dolistloop{\l@dcsnotetext}%
5781   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5782 %

```

And we do the same for left and right notes (not movable).

```

5783   \gdef\@templ@d{}%
5784   \gdef\@templ@n{\l@dcsnotetext\l@dcsnotetext\l@dcsnotetext@r}%
5785   \ifx\@templ@d\@templ@n \else%
5786     \if@twocolumn%

```

```

5787 \if@firstcolumn%
5788 \setl@dlp@rbox{##1}{\sidenotecontent@}%
5789 \else%
5790 \setl@drp@rbox{\sidenotecontent@}%
5791 \fi%
5792 \else%
5793 \@l@dttempcntb=\sidenote@margin%
5794 \ifnum\@l@dttempcntb>\@ne%
5795 \advance\@l@dttempcntb by\page@num%
5796 \fi%
5797 \ifodd\@l@dttempcntb%
5798 \setl@drp@rbox{\sidenotecontent@}%
5799 \gdef\sidenotecontent@{}%
5800 \numgdef{\itemcount@}{0}%
5801 \dolistloop{\l@dcsnotetext@l}%
5802 \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5803 \setl@dlp@rbox{\sidenotecontent@}%
5804 \else%
5805 \setl@dlp@rbox{\sidenotecontent@}%
5806 \gdef\sidenotecontent@{}%
5807 \numgdef{\itemcount@}{0}%
5808 \dolistloop{\l@dcsnotetext@r}%
5809 \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5810 \setl@drp@rbox{\sidenotecontent@}%
5811 \fi%
5812 \fi%
5813 \fi%
5814 }
5815 %

```

## XXV Minipages and such

We can put footnotes into minipages. The preparatory code has been set up earlier, all that remains is to ensure that it is available inside a minipage box. This requires some alteration to the kernel code, specifically the `\@iiiminipage` and `\endminipage` macros. We will arrange this so that additional series can be easily added.

`\l@dfheetbeginmini` These will be the hooks in `\@iiiminipage` and `\endminipage`.  
`\l@dfheetendmini` They can be extended to handle other things if necessary.

```

5816 \ifnoledgroup@\else%
5817 \newcommand*{\l@dfheetbeginmini}{\@ledgrouptrue\l@dedbeginmini\l@dfambeginmini}
5818 \newcommand*{\l@dfheetendmini}{%
5819 \IfStrEq{critical-familiar}{\@mpfnpos}%
5820 {\l@dedendmini\l@dfamendmini}%
5821 {%
5822 \IfStrEq{familiar-critical}{\@mpfnpos}%
5823 {\l@dfamendmini\l@dedendmini}%

```

```

5824         {\l@dedendmini\l@dfamendmini}%
5825     }%
5826 }%
5827 %

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.

`\l@dedendmini`

```

5828 \newcommand*{\l@dedbeginmini}{%
5829   \unless\ifnocritical@%
5830     \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5831     \dolistloop{\@series}%
5832   \fi%
5833 }
5834 \newcommand*{\l@dedendmini}{%
5835   \unless\ifnocritical@%
5836     \ifl@dpairing%
5837       \ifledRcol%
5838         \flush@notesR%
5839       \else%
5840         \flush@notes%
5841       \fi%
5842     \fi
5843     \def\do##1{%
5844       \ifvoid\csuse{mp##1footins}\else%
5845         \ifl@dpairing\ifparledgroup%
5846           \ifledRcol%
5847             \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
5848 skip\@nameuse{mp##1footins}}%
5849           \else%
5850             \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
5851 +\skip\@nameuse{mp##1footins}}%
5852           \fi%
5853         \fi\fi%
5854         \csuse{mp##1footgroup}{##1}%
5855       \fi}%
5856     \dolistloop{\@series}%
5857   \fi%
5858 }%

```

`\l@dfambeginmini` These handle the initiation and closure of familiar footnotes in a minipage environment.

`\l@dfamendmini`

```

5859 \newcommand*{\l@dfambeginmini}{%
5860   \unless\ifnofamiliar@%
5861     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5862     \dolistloop{\@series}%
5863   \fi%
5864 }%
5865 %

```

```

5866 \newcommand*{\l@dfamendmini}{%
5867 \unless\ifnofamiliar@%
5868 \def\do##1{%
5869 \ifvoid\csuse{mpfootins##1}\else%
5870 \csuse{mpfootgroup##1}{##1}%
5871 \fi}%
5872 \dolistloop{\@series}%
5873 \fi%
5874 }%
5875 %

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

5876 \patchcmd%
5877 {\@iiiminipage}%
5878 {\let\@footnotetext\@mpfootnotetext}%
5879 {\let\@footnotetext\@mpfootnotetext\l@dfetbeginmini}%
5880 {}%
5881 {\led@error@fail@patch@\@iiiminipage}%
5882 %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5883 \patchcmd%
5884 {\endminipage}%
5885 {\footnoterule}%
5886 {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5887 {}%
5888 {\led@error@fail@patch@endminipage}
5889
5890 \patchcmd%
5891 {\endminipage}%
5892 {\@minipagefalse}%
5893 {\l@dfetendmini\@minipagefalse}%
5894 {}%
5895 {\led@error@fail@patch@endminipage}
5896
5897 %

```

`\l@dunboxmpfoot` `\l@dunboxmpfoot` insert normal footnotes for `ledgroup`.

`edgroup@beforenormalnotes`

```

5898 \newcommand*{\l@dunboxmpfoot}{%
5899 \vskip\skip\@mpfootins
5900 \normalcolor
5901 \footnoterule
5902 \l@advance@parledgroup@beforenormalnotes
5903 \unvbox\@mpfootins%
5904 }
5905 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5906 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5907   \ifparledgroup
5908     \ifl@pairing
5909       \ifledRcol
5910         \dimdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@mpfootins}
5911       \else
5912         \dimdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+
skip\@mpfootins}
5913       \fi
5914     \fi
5915   \fi
5916 }
5917 %

```

`ledgroup` This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, fixed width minipage.

```

5918
5919 \newenvironment{ledgroup}{%
5920   \resetprevpage@num%
5921   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5922   \let\@footnotetext\@mpfootnotetext
5923   \l@dfetbeginmini%
5924 }{%
5925   \par
5926   \unskip
5927   \ifvoid\@mpfootins\else
5928     \l@dunboxmpfoot
5929   \fi
5930   \l@dfetendmini%
5931   \@ledgroupfalse%
5932 }
5933
5934
5935 %

```

`ledgroupsize` `\begin{ledgroupsize}[\langle pos \rangle]{\langle width \rangle}`

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable `\langle width \rangle` minipage. The optional `\langle pos \rangle` controls the sideways position of numbered text.

```

5936 \newenvironment{ledgroupsize}[2][1]{%
5937 %

```

Set the various text measures.

```

5938 \hsize #2\relax
5939 %

Initialize fills for centering.

5940 \let\ledllfill\hfil
5941 \let\ledrlfill\hfil
5942 \def\@tempa{#1}\def\@tempb{1}%
5943 %

Left adjusted numbered lines

5944 \ifx\@tempa\@tempb
5945 \let\ledllfill\relax
5946 \else
5947 \def\@tempb{r}%
5948 \ifx\@tempa\@tempb
5949 %

Right adjusted numbered lines

5950 \let\ledrlfill\relax
5951 \fi
5952 \fi
5953 %

Set up the footnoting.

5954 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5955 \let\@footnotetext\@mpfootnotetext
5956 \l@dfetbeginmini%
5957 }{f%
5958 \par
5959 \unskip
5960 \ifvoid\@mpfootins\else
5961 \l@dunboxmpfoot
5962 \fi
5963 \l@dfetendmini%
5964 }
5965 %
5966 %

Close the \ifnoledgroup@else.

5967 \fi%
5968 %

```

`\ifledgroupnotesL@` These boolean tests check if we are in the notes of a ledgroup. If we are, we do not  
`\ifledgroupnotesR@` number the lines. It could be useful for parallel ledgroup of `reledpar`.

```

5969 \newif\ifledgroupnotesL@
5970 \newif\ifledgroupnotesR@
5971 %

```

## XXVI Indexing

Here is some code for indexing using page and line numbers.

### XXVI.1 Looking on package order

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.

```

5972 \AtBeginDocument{%
5973   \unless\ifl@imakeidx%
5974     \ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5975   \fi%
5976   \unless\ifl@indextools%
5977     \ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5978   \fi%
5979 }
5980 %

```

### XXVI.2 Auxiliary macros for `\edindex`

`\pagelinesep` In order to get a correct line number we have to use the label/ref mechanism. These macros are for that.

```

\edindexlab
\c@labidx
5981 \newcommand{\pagelinesep}{-}
5982 \newcommand{\edindexlab}{$&}
5983 \newcounter{labidx}
5984 \setcounter{labidx}{0}
5985
5986 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5987 \newcommand{\doedindexlabel}{%
5988   \stepcounter{labidx}%
5989   \edlabel{\edindexlab\thelabidx}%
5990 }
5991
5992 %

```

`\thepageline` This macro makes up the page/line number combo from the label/ref. The associated counter is never directly used, but it is required in order to not have any error message with `\edgls`.

```

5993 \newcounter{pageline}%
5994 \renewcommand{\thepageline}{%
5995   \thepage%
5996   \pagelinesep%
5997   \xlineref{\edindexlab\thelabidx}%
5998 }
5999 %

```



`\thestartpageline` `\theendpageline` These macros make up the page/line start/end number when the `\edindex` command is called in critical notes.

```

6000 \newcommand{\thestartpageline}{%
6001   \l@dparsedstartpage%
6002   \pagelinesep%
6003   \l@dparsedstartline%
6004 }
6005 \newcommand{\theendpageline}{%
6006   \l@dparsedendpage%
6007   \pagelinesep%
6008   \l@dparsedendline%
6009 }
6010 %

```

### XXVI.3 Code specific to `\edindex` in critical footnotes

`\if@edindex@fornote@true` This boolean test is switching at the beginning of each critical note, to allow index referring to this note.

```

6011 \newif\if@edindex@fornote@
6012 %

```

`\prepare@edindex@fornote` This macro is called at the beginning of each critical note. It switches some parameters, to allow index referring to this note, with reference to page and line number. It also defines `\@ledinnote@command` which will be printed as an encapsulating command after the `|`.

```

6013 \newcommand{\prepare@edindex@fornote}[1]{%
6014   \l@dp@rsefootspec#1|%
6015   \@edindex@fornote@true%
6016 }
6017 %

```

`\edindex@ledinnote@command` The `\get@edindex@ledinnote@command` macro defines a `\@ledinnote@command` command which is added as an attribute (text inserted after `|`) of the next index entry.

Consequently, we write the definition of the location reference attribute in the `.xdy` file.

```

6018 \newcommand{\get@edindex@ledinnote@command}{%
6019   \ifxindy@%
6020     \gdef\@ledinnote@command{%
6021       ledinnote\thelabidx%
6022     }%
6023     \ifxindyhyperref@%
6024       \immediate\write\eledmac@xindy@out{%
6025         (define-attributes ("ledinnote\thelabidx"))^^J
6026         \space\space(markup-locref^^J
6027         \eledmacmarkuplocrefdepth^^J

```

```

6028     :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
}{"^^J
6029     :close "}"^^J
6030     :attr "ledinnote\thelabidx"^^J
6031     )
6032   }%
6033   \else%
6034     \immediate\write\eledmac@xindy@out{%
6035     (define-attributes ("ledinnote\thelabidx"))^^J
6036     \space\space(markup-locref^^J
6037     \eledmacmarkuplocrefdepth^^J
6038     :open "\string\ledinnote{\@index@command}"^^J
6039     :close "}"^^J
6040     :attr "ledinnote\thelabidx"^^J
6041     )
6042   }%
6043   \fi%
6044   %

```

If we do not use xindy option, `\@ledinnote@command` will produce something like `ledinnote{formattingcommand}`.

```

6045   \else%
6046     \gdef\@ledinnote@command{%
6047     ledinnote[\edindexlab\thelabidx]{\@index@command}%
6048   }%
6049   \fi%
6050 }
6051 %

```

#### XXVI.4 Analysis of command in indexed text

`\get@index@command` This macro is used to analyze if a text to be indexed has a command after a |.

```

6052 \def\get@index@command#1|#2+{%
6053   \gdef\@index@txt{#1}%
6054   \gdef\@index@command{#2}%
6055   \xdef\@index@parenthesis{}%
6056   \IfBeginWith{\@index@command}{(}{%
6057     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
6058     \global\let\@index@command\@index@command@%
6059     \xdef\@index@parenthesis{(%}%
6060   }{}%
6061   \IfBeginWith{\@index@command}{)}{%
6062     \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
6063     \global\let\@index@command\@index@command@%
6064     \xdef\@index@parenthesis{)%}%
6065   }{}%
6066 }
6067 %

```

## XXVI.5 Code for the formatted index

`\ledinnote` These macros are used to specify that an index reference points to a note. Arguments of `\ledinnote` are: #1 (optional): the label for the hyperlink, #2: command applied to the number, #3: the number itself.

```

6068 \newcommandx{\ledinnote}[3][1,usedefault]{%
6069   \ifboolexpr{%
6070     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
6071     or%
6072     bool {xindyhyperref@}%
6073   }%
6074   {%
6075     \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
6076   }%
6077   {%
6078     \csuse{#2}{\ledinnotemark{#3}}%
6079   }%
6080 }%
6081 \newcommand{\ledinnotehyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
6082 {#2}}}}%
6083 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
6084 %

```

## XXVI.6 Main code

Eledmac and ledmac were using the specific indexing tools of the memoir in order to allow multiple index. However, eledmac used imakeidx or indextools tools when one these two package was loaded. This system forced to maintained a double code, which was not very useful. Since reledmac, we use only the imakeidx or indextools tools.

The memoir class provides more flexible indexing than the standard classes. We need different code if the memoir class is being used, except if imakeidx or indextools is used.

`\edindex` Write the index information to the idx file.  
`\@wredindex`

```

6084 \newcommandx{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
index name, #2 = the text
6085   \ifl@imakeidx%
6086     \if@edindex@fornote@%
6087     \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
6088     \get@edindex@ledinnote@command%
6089     \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
}{\thestartpageline}%
6090     \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
}{\theendpageline}%
6091     \else%
6092     \get@edindex@hyperref{#2}%
6093     \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%

```

```

6094 \fi%
6095 \else%
6096 \if@edindex@fornote@%
6097 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
6098 \get@edindex@ledinnote@command%
6099 \expandafter\protected@write\@indexfile{%
6100 {\string\indexentry{\@index@txt|(\@ledinnote@command){\thestartpageline}
6101 }%
6102 \expandafter\protected@write\@indexfile{%
6103 {\string\indexentry{\@index@txt|)\@ledinnote@command){\theendpageline}
6104 }%
6105 \else%
6106 \protected@write\@indexfile{%
6107 {\string\indexentry{#2}{\thepageline}
6108 }%
6109 \fi%
6110 \fi%
6111 \endgroup
6112 \@esphack%
6113 }
6114 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

6115 \pretocmd{\makeindex}{%
6116 \def\edindex{\@bsphack
6117 \doedindexlabel
6118 \begingroup
6119 \@sanitize
6120 \@wredindex}}{}{}
6121 \newcommand{\edindex}[1]{\@bsphack\@esphack}
6122 %

```

## XXVI.7 Hyperlink

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

6123 \newcommand{\hyperlinkformat}[3]{%
6124 \ifstrempy{#1}%
6125 {\hyperlink{#2}{#3}}%
6126 {\csuse{#1}{\hyperlink{#2}{#3}}%
6127 }}
6128 %

```

`\hyperlinkR` `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.

```

6129 \newcommand{\hyperlinkR}[2]{%

```

```

6130 \hyperlink{#1}{#2\@Rlineflag}%
6131 }%
6132
6133 %

```

`\hyperlinkformatR` `\hyperlinkformatR` command is to be used to create a internal hyperlink, a format and a `\@Rlineflag`, when indexing.

```

6134 \newcommand{\hyperlinkformatR}[3]{%
6135   \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
6136 }%
6137
6138 %

```

`\get@edindex@hyperref` `\get@edindex@hyperref` is to be used to define the `\@edindex@hyperref` macro, which, in index, links to the point where the index was called (with `hyperref`).

```

6139 \newcommand{\get@edindex@hyperref}[1]{%
6140 %

```

We have to disable temporary spaces to work through a `xstring` bug (or feature?)

```

6141 \edef\temp@{%
6142   \catcode`\ =9 %space need for catcode
6143   \detokenize{#1}%For active character in unicode
6144   \catcode`\ =10 % space need for catcode
6145 }%
6146 %

```

Now, we define `\@edindex@hyperref` if the `hyperindex` of `hyperref` is enabled.

```

6147 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
6148   \IfSubStr{\temp@}{|}%
6149   {\get@index@command#1+%
6150   \ifledRcol%
6151     \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
6152     hyperlinkformatR{\@index@command}%
6153     {\edindexlab\thelabidx}}%
6154   \else%
6155     \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
6156     hyperlinkformat{\@index@command}%
6157     {\edindexlab\thelabidx}}%
6158   \fi%
6159 }%
6160 {\get@index@command#1|+%
6161   \ifledRcol%
6162     \gdef\@edindex@hyperref{|\hyperlinkR{\edindexlab\thelabidx}}%
6163   \else%
6164     \gdef\@edindex@hyperref{|\hyperlink{\edindexlab\thelabidx}}%
6165   \fi%
6166 }%
6167 }%
6168 %

```

```

6169 % If we use both xindy and hyperref, first get the \protect\cs{
      index@command} command.
6170 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
6171 %   \begin{macrocode}
6172     {\ifxindyhyperref%,
6173      \IfSubStr{\temp@}{|}%
6174       {\get@index@command#1+}%
6175       {\get@index@command#1|+}%
6176      \gdef\@edindex@hyperref{|eledmac\thelabidx}%
6177 %

```

If we start a reference range by a opening parenthesis, store the `\thelabidx` for the current `\edindex`, then define `\@edindex@hyperref` in the form `| (eledmac\thelabidx`.

```

6178     \IfStrEq{\@index@parenthesis}{(}%
6179     {%
6180     \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
6181     \gdef\@edindex@hyperref{| (eledmac\thelabidx}%
6182     }%
6183     }%
6184 %

```

This `\thelabidx` will be called back at the closing parenthesis, to have the same number in `\@edindex@hyperref` command that we had at the opening parenthesis. `\@edindex@hyperref` start by a closing parenthesis, then followed by `eledmacXXX` where `XXX` is the `\thelabidx` of the opening `\edindex`.

```

6185     \IfStrEq{\@index@parenthesis}{)}%
6186     {%
6187     \xdef\@edindex@hyperref{|)eledmac\csuse{xindyparenthesis@\@index@txt}}%
6188     \global\csundef{xindyparenthesis@\@index@txt}%
6189     }%
6190 %

```

Write in the `.xdy` file the attributes of the location.

```

6191     {%
6192     \immediate\write\eledmac@xindy@out{%
6193     (define-attributes ("eledmac\thelabidx"))^^J
6194     \space\space(markup-locref^^J
6195     \eledmacmarkuplocrefdepth^^J
6196     :open "\string\hyperlink%
6197     \ifledRcol R\fi%
6198     {\edindexlab\thelabidx}%
6199     {\ifdefempty{\@index@command}%
6200     }%
6201     {\@backslashchar\@index@command}%
6202     {"^^J
6203     :close "}")^^J
6204     :attr "eledmac\thelabidx"^^J
6205     )

```

```

6206     }%
6207     }%
6208 %

```

And now, in any other case.

```

6209     \else%
6210         \gdef\@index@txt{#1}%
6211         \gdef\@edindex@hyperref{}%
6212     \fi%
6213     }%
6214 }
6215 %

```

## XXVI.8 ‘innote’ and ‘notenumber’ option of *indextols* package

`\led@set@index@fornote` The `\led@set@index@fornote` is called when a familiar footnote is inserted — and not when it is read — and changes the `\index` command depending of the option of the *indextols* package. Its only argument is the note series.

```

6216 \newcommand{\led@set@index@fornote}[1]{%
6217     \ifbool{indtl@innote}%
6218         {\let\index\nindex}%
6219         {}%
6220     \ifbool{indtl@notenumber}%
6221         {%
6222         \renewcommand{\index}[2][\indtl@jobname]{%
6223             \orig@@index[##1]{%
6224                 ##2|innotenumber{\csuse{thefootnote#1}}%
6225             }%
6226         }%
6227     }%
6228     {}%
6229 }%
6230 %

```

`\led@reinit@index@fornote` The `\led@reinit@index@fornote` just reset the default value of `\index`.

```

6231 \newcommand{\led@reinit@index@fornote}{%
6232     \ifbool{indtl@innote}%
6233         {\let\index\orig@@index}%
6234         {}%
6235     \ifbool{indtl@notenumber}%
6236         {\let\index\orig@@index}%
6237         {}%
6238 }%
6239 %

```

## XXVII Glossaries

Here, we define the \gls-like commands prefixed by ed, only if the package `glossaries` is loaded.

```
6240 \AtBeginDocument{%
6241   \@ifpackageloaded{glossaries}{%
6242     %
```

First those which arguments are [*options*]{*label*}[*insert*].

```
6243   \renewcommand{\do}[1]{%
6244     \expandafter\DeclareRobustCommand\csname ed#1\endcsname[3][1,3,
usedefault]{%
6245       \doedindexlabel%
6246       \csname#1\endcsname[counter=pageline,##1]{##2}[##3]%
6247     }%
6248     \expandafter\WithSuffix\expandafter\DeclareRobustCommand\csname ed
#1\endcsname*[3][1,3,usedefault]{%
6249       \doedindexlabel%
6250       \csname#1\endcsname*[counter=pageline,##1]{##2}[##3]%
6251     }%
6252   }%
6253   \docsvlist{%
6254     gls,%
6255     Gls,%
6256     GLS,%
6257     glspl,%
6258     Glspl,%
6259     GLSpl,%
6260     glstext,%
6261     Glstext,%
6262     GLStext,%
6263     Glsfirst,%
6264     GLSfirst,%
6265     glsplural,%
6266     Glsplural,%
6267     GLSplural,%
6268     glsfirstplural,%
6269     Glsfirstplural,%
6270     GLSfirstplural,%
6271     glsname,%
6272     Glsname,%
6273     GLSname,%
6274     glssymbol,%
6275     Glsymbol,%
6276     GLSsymbol,%
6277     glsdesc,%
6278     Glsdesc,%
6279     GLSdesc,%
6280     glsuseri,%
```



```

6281     Glsuseri,%
6282     GLSuseri,%
6283     glsuserii,%
6284     Glsuserii,%
6285     GLSuserii,%
6286     glsuseriii,%
6287     Glsuseriii,%
6288     GLSuseriii,%
6289     glsuseriv,%
6290     Glsuseriv,%
6291     GLSuseriv,%
6292     glsuserv,%
6293     Glsuserv,%
6294     GLSuserv,%
6295     glsuservi,%
6296     Glsuservi,%
6297     GLSuservi%
6298     }%
6299 %

```

First those which arguments are [*options*]{*label*}{*link text*}.

```

6300     \renewcommand{\do}[1]{%
6301         \expandafter\DeclareRobustCommand\csname ed#1\endcsname[3][1,
usedefault]{%
6302             \doedindexlabel%
6303             \csname#1\endcsname[counter=pageline,##1]{##2}{##3}%
6304         }%
6305         \expandafter\WithSuffix\expandafter\DeclareRobustCommand\csname ed
#1\endcsname*[3][1,usedefault]{%
6306             \doedindexlabel%
6307             \csname#1\endcsname*[counter=pageline,##1]{##2}{##3}%
6308         }%
6309     }%
6310     \docsvlist{glsdisp,glslink}%
6311 }{%
6312 }%
6313 %

```

## XXVIII Verse

The original code is principally Wayne Sullivan's code from `edstanza`. However, the code has been many time modified by Maïeul Rouquette in order to obtain new features and improved compatibility with `reledpar`.

### XXVIII.1 Hanging symbol management

`\@hangingsymbol` The macro `\@hangingsymbol` is used to insert a symbol on each hanging of verses. It is set by user level macro `\sethangingsymbol`.  
`\ifinstanza`

For example, in french typographie the symbol is ‘[’. We obtain it by the next code:

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

The `\ifinstanza` boolean is used to be sure that we are in a stanza part.

```
6314 \def\@hangingsymbol{}
6315 \newcommand*\sethangingsymbol}[1]{%
6316   \gdef\@hangingsymbol{#1}%
6317 }%
6318 \newif\ifinstanza
6319 %
```

`\inserthangingsymbol` The boolean `\ifinserthangingsymbol` is set to TRUE when `\@lock` is greater than 1, i.e. when we are not in the first line of a verse. The switch of `\ifinserthangingsymbol` is made in `\do@line` before the printing of line but after the line number calculation.

```
6320 \newif\ifinserthangingsymbol
6321 \newcommand\inserthangingsymbol{%
6322   \ifinserthangingsymbol%
6323     \ifinstanza%
6324       \@hangingsymbol%
6325     \fi%
6326 \fi%
6327 }
6328 %
```

## XXVIII.2 Using & character

`\ampersand` Within a stanza the `\&` macro is going to be usurped. We need an alias in case an `&` needs to be typeset in a stanza. Define it rather than letting it in case some other package has already defined it.

```
6329 \newcommand*\ampersand}{\char`\&}
6330
6331 %
```

## XXVIII.3 Code category setting

`\stanza@count` Before we can define the main macros we need to save and reset some category codes.  
`\stanzaindentbase` To save the current values we use `\next` and `\body` from the `\loop` macro.

```
6332 \chardef\body=\catcode`\@
6333 \catcode`\@=11
6334 \chardef\next=\catcode`\&
6335 \catcode`\&=\active
6336
6337 %
```

## XXVIII.4 Stanza count and indent

A count register is allocated for counting lines in a stanza; also allocated is a dimension register which is used to specify the base value for line indentation; all stanza indentations are multiples of this value. The default value of `\stanzaindentbase` is 20pt.

```
6338 \newcount\stanza@count
6339 \newlength{\stanzaindentbase}
6340 \setlength{\stanzaindentbase}{20pt}
6341
6342 %
```

`\strip@szacnt` The indentations of stanza lines are non-negative integer multiples of the unit called `\stanzaindentbase`. To make it easier for the user to specify these numbers, some list macros are defined. These take numerical values in a list separated by commas and assign the values to special control sequences using `\mathchardef`. Though this does limit the range from 0 to 32767, it should suffice for most applications, including *penalties*, which will be discussed below.

`\setstanzavalues`

```
6343 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
6344 \newcommand*\setstanzavalues}[2]{\def\@tempa{#2,,|}%
6345   \stanza@count\z@
6346   \def\next{\expandafter\strip@szacnt\@tempa
6347     \ifx\@tempb\empty\let\next\relax\else
6348     \expandafter\mathchardef\csname #1@\number\stanza@count
6349     \@endcsname\@tempb\relax
6350     \advance\stanza@count\@ne\fi\next}%
6351   \next}
6352
6353 %
```

`\setstanzaindents` In the original edmac, `\setstanzavalues{sza}{⟨...⟩}` had to be called to set the indents, and similarly `\setstanzavalues{szp}{⟨...⟩}` to set the penalties. `\setstanzaindents` and `\setstanzapenalties` macros are a convenience to give the user one less thing to worry about (misspelling the first argument).

`\setstanzapenalties`

```
6354 \newcommand*\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
6355 \newcommand*\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
6356 %
6357 %
```

`\managestanza@modulo` Since version 0.13, the `stanzaindentsrepetition` counter can be used when the indentation is repeated every *n* verses. The `\managestanza@modulo` is a command which modifies the counter `stanza@modulo`. The command adds 1 to `stanza@modulo`, but if `stanza@modulo` is equal to the `stanzaindentsrepetition` counter, the command restarts it.

```
6358 \newcounter{stanzaindentsrepetition}
6359 \newcount\stanza@modulo
6360
```

```

6361 \newcommand*\managestanza@modulo}[0]{%
6362   \advance\stanza@modulo\@ne%
6363   \ifnum\stanza@modulo>\value{stanzaindentsrepetition}%
6364     \stanza@modulo\@ne%
6365   \fi%
6366 }
6367 %

```

**\stanzaindent** The macro `\stanzaindent`, when called at the beginning of a verse, changes the indentation normally defined for this verse by `\setstanzaindent`. The starred version **\stanzaindent\*** skips the current verse for the repetition of stanza indent.

```

6368 \newcommand{\stanzaindent}[1]{%
6369   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
6370   \ignorespaces%
6371 }%
6372 \WithSuffix\newcommand\stanzaindent*[1]{%
6373   \stanzaindent{#1}%
6374   \global\advance\stanza@modulo-\@ne%
6375   \ifnum\stanza@modulo=0%
6376     \global\stanza@modulo=\value{stanzaindentsrepetition}%
6377   \fi%
6378   \ignorespaces%
6379 }%
6380 %

```

## XXVIII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```

\thestanza51 \newcounter{stanza}
6382 \renewcommand{\thestanza}{%
6383   \textbf{\arabic{stanza}}%
6384 }
6385 %

```

**\ifnumberstanza** Then, macro to activate automatically numbering of stanza.

```

6386 \newif\ifnumberstanza%
6387 %

```

**\@insertstanzanumber** Now, macro called at the first line of of verse of a stanza.

```

6388 \newcommand{\@insertstanzanumber}[0]{%
6389   \ifnumberstanza%
6390     \ifl@dpairing%
6391       \ifledRcol%
6392         \stanzanumwrapper{\thestanzaR}%
6393       \else%

```

```

6394     \stanzanumwrapper{\thestanzaL}%
6395     \fi%
6396   \else%
6397     \stanzanumwrapper{\thestanza}%
6398     \fi%
6399     \setline{1}%
6400   \fi%
6401 }%
6402 %

```

`\@advancestanza` Also a command to advance the counter of stanza.

```

6403 \newcommand{\@advancestanza}[0]{%
6404   \ifnumberstanza%
6405     \ifl@pairing%
6406       \ifledRcol%
6407         \addtocounter{stanzaR}{1}%
6408       \else%
6409         \addtocounter{stanzaL}{1}%
6410       \fi%
6411     \else%
6412       \addtocounter{stanza}{1}%
6413     \fi%
6414   \fi%
6415 }%
6416 %

```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```

6417 \newcommand{\stanzanumwrapper}[1]{%
6418   \flagstanza{#1}%
6419 }%
6420 %

```

## XXVIII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

6421 \newcommand{\printstanza}[0]{%
6422   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
6423     l@dprintingcolumns}}{%
6424     \ifledRcol@%
6425       \thestanzaR%
6426     \else%
6427       \thestanzaL%
6428     \fi%
6429   }{%
6430     \thestanza%
6431   }%

```

```
6431 }
6432 %
```

### XXVIII.7 Main work

`\stanza@line` Now we arrive at the main works. `\stanza@line` sets the indentation for the line and `\stanza@hang` starts a numbered paragraph—each line is treated as a paragraph. `\stanza@hang` sets the hanging indentation to be used if the stanza line requires more than one print line.

`\sza@penalty` If it is known that each stanza line will fit on one print line, it is advisable to set the hanging indentation to zero. `\sza@penalty` places the specified penalty following each stanza line. By default, this facility is turned off so that no penalty is included. However, the user may initiate these penalties to indicate good and bad places in the stanza for page breaking.

```
6433 \newcommandx{\stanza@line}[1][1]{
6434   \ifnum\value{stanzaindentsrepetition}=0
6435     \parindent=\csname sza@\number\stanza@count
6436       @\endcsname\stanzaindentbase
6437   \else
6438     \parindent=\csname sza@\number\stanza@modulo
6439       @\endcsname\stanzaindentbase
6440   \managestanza@modulo
6441   \fi
6442   \pstart[#1]\stanza@hang\ignorespaces}
6443 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
6444   \hangindent\expandafter
6445   \noexpand\csname sza@0@\endcsname\stanzaindentbase
6446   \hangafter\@ne}
6447 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
6448   \ifnum\count@>@M\advance\count@-\@M\penalty-\else
6449   \penalty\fi\count@}
6450 %
```

`\@startstanza` Now we have the components of the `\stanza` macro, which appears at the start of a group of lines. This macro initializes the count and checks to see if hanging indentation and penalties are to be included. Hanging indentation suspends the line count, so that the enumeration is by verse line rather than by print line. If the print line count is desired, invoke `\let\startlock\relax` and do the same for `\endlock`. Here and above we have used `\xdef` to make the stored macros take up a bit less space, but it also makes them more obscure to the reader. Lines of the stanza are delimited by ampersands `&`. The last line of the stanza must end with `\&`.

```
6451 \xdef\@startstanza[#1]{%
6452   \noexpand\instanzatrue\expandafter
6453   \begingroup%
6454   \catcode`\noexpand\&\active%
6455   \global\stanza@count\@ne\stanza@modulo\@ne
6456   \noexpand\ifnum\expandafter\noexpand
```

```

6457 \csname sza@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
6458 \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
6459 \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
6460 \expandafter\noexpand\csname szp@0@\endcsname=\z@
6461 \let\noexpand\sza@penalty\relax\noexpand\fi%
6462 \def\noexpand&{%
6463   \noexpand\newverse [] []}%
6464 \def\noexpand\&{\noexpand\@stopstanza}%
6465 \noexpand\@advancestanza\@number%
6466 \noexpand\stanza@line[#1]\noexpand\@insertstanza\@number%
6467 \let\par\relax\ignorespaces%No paragraph in verses
6468 }
6469
6470 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
6471
6472 \newcommandx{\@stopstanza}[1][1,usedefault]{%
6473   \unskip%
6474   \endlock%
6475   \pend[#1]%
6476   \endgroup%
6477   \instanzafalse%
6478 }
6479
6480 \newcommandx*\@newverse}[2][1,2,usedefault]{%
6481   \unskip%
6482   \endlock\pend[#1]\sza@penalty\global%
6483   \advance\stanza@count\@ne\stanza@line[#2]%
6484 }
6485
6486 %

```

**\flagstanza** Use `\flagstanza[len]{text}` at the start of a line to put *text* a distance *len* before the start of the line. The default for *len* is `\stanzaindentbase`.

```

6487 \newcommand*\flagstanza}[2][\stanzaindentbase]{%
6488   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
6489
6490 %

```

## XXVIII.8 Restore catcode and penalties

The ampersand `&` is used to mark the end of each stanza line, except the last, which is marked with `\&`. This means that `\halign` may not be used directly within a stanza line. This does not affect macros involving alignments defined outside `\stanza \&`. Since these macros usurp the control sequence `\&`, the replacement `\ampersand` is defined to be used if this symbol is needed in a stanza. Also we reset the modified category codes and initialize the penalty default.

```

6491 \catcode`\&=\next

```

```

6492 \catcode`\@=\body
6493 \setstanzavalues{szp}{0}
6494
6495 %

```

## XXIX Arrays and tables

### XXIX.1 Preamble: macro as environment

The following is borrowed, and renamed, from the `amsmath` package. See also the CTT thread ‘`eeq` and `amstex`’, 1995/08/31, started by Keith Reckdahl and ended definitively by David M. Jones.

Several of the `[math]` macros scan their body twice. This means we must collect all text in the body of an environment form before calling the macro.

`\@emptytoks` This is actually defined in the `amsgen` package.

```

6496 \newtoks\@emptytoks
6497
6498 %

```

The rest is from `amsmath`.

`\l@denvbody` A token register to contain the body.

```

6499 \newtoks\l@denvbody
6500
6501 %

```

`\addtol@denvbody` `\addtol@denvbody{arg}` adds `arg` to the token register `\l@denvbody`.

```

6502 \newcommand{\addtol@denvbody}[1]{%
6503   \global\l@denvbody\expandafter{\the\l@denvbody#1}}
6504
6505 %

```

`\l@dcollect@body` The macro `\l@dcollect@body` starts the scan for the `\end{env}` command of the current environment. It takes a macro name as argument. This macro is supposed to take the whole body of the environment as its argument. For example, given `cenv#1{...}` as a macro that processes `#1`, then the environment form, `\begin{env}` would call `\l@dcollect@body\cenv`.

```

6506 \newcommand{\l@dcollect@body}[1]{%
6507   \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}%
6508   \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currentvir}}%
6509   \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
6510   \begingroup
6511   \expandafter\let\csname\@currentvir\endcsname\l@dcollect@@body

```



```

6512 \edef\processl@denbody{\expandafter\noexpand\csname\@currentenv\
endcsname}%
6513 \processl@denbody%
6514 }%
6515
6516 %

```

**\l@dpush@begins** When adding a piece of the current environment's contents to \l@denbody, we scan it to check for additional \begin tokens, and add a 'b' to the stack for any that we find.

```

6517 \def\l@dpush@begins#1\begin#2{%
6518 \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
6519
6520 %

```

**\l@dcollect@@body** \l@dcollect@@body takes two arguments: the first will consist of all text up to the next \end command, and the second will be the \end command's argument. If there are any extra \begin commands in the body text, a marker is pushed onto a stack by the \l@dpush@begins function. Empty state for this stack means we have reached the \end that matches our original \begin. Otherwise we need to include the \end and its argument in the material we are adding to the environment body accumulator.

```

6521 \def\l@dcollect@@body#1\end#2{%
6522 \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
6523 \expandafter\@gobble\l@dbegin@stack}%
6524 \ifx\@empty\l@dbegin@stack
6525 \endgroup
6526 \@checkend{#2}%
6527 \addtol@denbody{#1}%
6528 \else
6529 \addtol@denbody{#1\end{#2}}%
6530 \fi
6531 \processl@denbody % A little tricky! Note the grouping
6532 }
6533
6534 %

```

There was a question on CTT about how to use \collect@body for a macro taking an argument. The following is part of that thread.

```

From: Heiko Oberdiek <oberdiek@uni-freiburg.de>
Newsgroups: comp.text.tex
Subject: Re: Using \collect@body with commands that take >1 argument
Date: Fri, 08 Aug 2003 09:03:20 +0200

```

eed132@psu.edu (Evan) wrote:

```

> I'm trying to make a new Latex environment that acts like the>
> \colorbox command that is part of the color package. I looked through
> the FAQ and ran across this bit about using the \collect@body command

```

```

> that is part of AMSLaTeX:
> http://www.tex.ac.uk/cgi-bin/texfaq2html?label=cmdasenv
>
> It almost works. If I do something like the following:
> \newcommand{\redbox}[1]{\colorbox{red}{#1}}
>
> \makeatletter
> \newenvironment{redbox}{\collect@body \redbox}{}
```

You will get an error message: Command `\redbox` already defined. Thus you must rename either the command `\redbox` or the environment name.

```

> \begin{coloredbox}{blue}
>   Yadda yadda yadda... this is on a blue background...
> \end{coloredbox}
> and can't figure out how to make the \collect@body take this.

> \collect@body \colorbox{red}
> \collect@body {\colorbox{red}}
```

The argument of `\collect@body` has to be one token exactly.

```

\documentclass{article}
\usepackage{color}
\usepackage{amsmath}

\newcommand{\redbox}[1]{\colorbox{red}{#1}}
\makeatletter
\newenvironment{coloredbox}[1]{%
  \def\next@{\colorbox{#1}}%
  \collect@body\next@
}{%

% ignore spaces at begin and end of environment
\newenvironment{coloredboxII}[1]{%
  \def\next@{\mycoloredbox{#1}}%
  \collect@body\next@
}{%
\newcommand{\mycoloredbox}[2]{%
  \colorbox{#1}{\ignorespaces#2\unskip}%
}

% support of optional color model argument
\newcommand\coloredboxIII\endcsname{}
\def\coloredboxIII#1#2{%
  \@coloredboxIII{#1}%
}
\def\@coloredboxIII#1#2{%
  \def\next@{\mycoloredboxIII{#1}{#2}}%
```

```

\collect@body\next@
}
\newcommand{\mycoloredboxIII}[3]{%
\colorbox#1{#2}{\ignorespaces#3\unskip}%
}

\makeatother

\begin{document}
Black text before
\begin{coloredbox}{blue}
Hello World
\end{coloredbox}
Black text after

Black text before
\begin{coloredboxII}{blue}
Hello World
\end{coloredboxII}
Black text after

Black text before
\begin{coloredboxIII}[rgb]{0,0,1}
Hello World
\end{coloredboxIII}
Black text after

\end{document}

Yours sincerely
Heiko <oberdiek@uni-freiburg.de>

```

## XXIX.2 Tabular environments

This is based on the work by Herbert Breger in developing `tabmac.tex`.

The original `tabmac.tex` file was void of comments or any explanatory text other than the above notice. The algorithm is Breger's. Peter Wilson have made some cosmetic changes to the original code and reimplemented some things so they are more LaTeX-like. All the commentary are from Peter Wilson, as are any mistake or errors.

However, Maïeul Rouquette has modified code in order to add new features of `eledmac` and `reledmac`.

### XXIX.2.1 Disabling and restoring commands

`\l@dtabnoexpands` More no expansion for critical and familiar footnotes in tabular environment.

```

6535 \newcommand*\l@dtabnoexpands}{%
6536 \let\rtab=0%

```

```

6537 \let\ctab=0%
6538 \let\ltab=0%
6539 \let\rtabtext=0%
6540 \let\ltabtext=0%
6541 \let\ctabtext=0%
6542 \let\edbeforetab=0%
6543 \let\edaftertab=0%
6544 \let\edatleft=0%
6545 \let\edatright=0%
6546 \let\edvertline=0%
6547 \let\edvertdots=0%
6548 \let\edrowfill=0%
6549 }
6550
6551 %

```

`\disable@famiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in edtabularx and edarrayx environments.

`\restore@famiarnotes`

```

6552 \newcommand{\disable@famiarnotes}{%
6553 \unless\ifnofamiliar%
6554 \def\do##1{%
6555 \csletcs{footnote@##1}{footnote##1}%
6556 \expandafter\renewcommand\csname footnote##1\endcsname[1]{%
6557 \protected@csxdef{@thefnmark##1}{\csuse{thefootnote##1}}%
6558 \csuse{@footnotemark##1}}%
6559 }%
6560 }%
6561 \dolistloop{\@series}%
6562 \fi%
6563 }%
6564 \newcommand{\restore@famiarnotes}{%
6565 \unless\ifnofamiliar%
6566 \def\do##1{%
6567 \csletcs{footnote##1}{footnote@##1}%
6568 }%
6569 \dolistloop{\@series}%
6570 \fi%
6571 }%
6572
6573 %

```

`\disable@sidenotes` The sames, for side notes.

`\restore@sidenotes`

```

6574 \newcommand{\disable@sidenotes}{%
6575 \let\@@ledrightnote\ledrightnote%
6576 \let\@@ledleftnote\ledleftnote%
6577 \let\@@ledsidenote\ledsidenote%
6578 \let\ledrightnote@gobble%
6579 \let\ledleftnote@gobble%

```

```

6580 \let\ledsidenote\@gobble%
6581 }%
6582 \newcommand{\restore@sidenotes}{%
6583 \let\ledrightnote\@ledrightnote%
6584 \let\ledleftnote\@ledleftnote%
6585 \let\ledsidenote\@ledsidenote%
6586 }%
6587 %

```

**\disable@notes** Disable/restore side and familiar notes.

```

\restore@notes
6588 \newcommand{\disable@notes}{%
6589 \disable@sidenotes%
6590 \disable@familiarnotes%
6591 }%
6592 \newcommand{\restore@notes}{%
6593 \restore@sidenotes%
6594 \restore@familiarnotes%
6595 }%
6596 %

```

**\EDTEXT** We need to be able to modify the `\edtext` macros and also restore their original definitions.

```

\edtext
6597 \let\EDTEXT=\edtext
6598 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
6599 %

```

**\EDLABEL** We need to be able to modify and restore the `\edlabel` macro.

```

\edlabel
6600 \let\EDLABEL=\edlabel
6601 \newcommand*\xedlabel[1]{\EDLABEL{#1}}
6602 %

```

**\EDINDEX** Macros supporting modification and restoration of `\edindex`.

```

\xedindex
\nullindex
6603 \let\EDINDEX=\edindex
6604 \newcommand{\xedindex}{\@bsphack%
6605 \ifnextchar [{\led@index}{\led@index[\jobname]}}
6606 \newcommand{\nullindex}[2][\jobname]{\@bsphack\@esphack}
6607
6608 %

```

**\@line@num** Macro supporting restoration of `\linenum`.

```

6609 \let\@line@num=\linenum
6610 %

```

**\l@dgobblearg** `\l@dgobbleoptarg[⟨arg⟩]{⟨arg⟩}` replaces these two arguments (first is optional) by `\relax`.

```

6611 \newcommand*{\l@dgobbleoptarg}[2][\relax]%
6612
6613 %

```

```

\Relax14 \let\Relax=\relax
\NEXT15 \let\NEXT=\next
6616
6617 %

```

**\l@dmodforedtext** Modify and restore various macros for when `\edtext` is used.  
**\l@drestoreforedtext**

```

6618 \newcommand{\l@dmodforedtext}{%
6619   \let\edtext\relax
6620   \def\do##1{\global\csletcs{##1footnote}{\l@dgobbleoptarg}}%
6621   \dolistloop{\@series}%
6622   \let\edindex\nulledindex
6623   \let\linenum\@gobble}
6624 \newcommand{\l@drestoreforedtext}{%
6625   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
6626   \dolistloop{\@series}%
6627   \let\edindex\xedindex}
6628 %

```

**\l@dnullfills** Nullify and restore some column fillers, etc.  
**\l@drestorefills**

```

6629 \newcommand{\l@dnullfills}{%
6630   \def\edlabel##1{%
6631     \def\edrowfill##1##2##3{%
6632     }
6633   \newcommand{\l@drestorefills}{%
6634     \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
6635   }
6636 %
6637 %

```

**\letsforverteilen** Gathers some lets and other code that is common to the `*verteilen*` macros.

```

6638 \newcommand{\letsforverteilen}{%
6639   \let\edtext\xedtext
6640   \let\edindex\xedindex
6641   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
6642   \dolistloop{\@series}%
6643   \let\linenum\@line@num
6644   \hilfsskip=\l@dcwidth%
6645   \advance\hilfsskip by -\wd\hilfsbox
6646   \def\edlabel##1{\xedlabel{##1}}
6647
6648 %

```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for  
`\enablel@dtabfeet` `\edtext`.

```
6649 \newcommand\disablel@dtabfeet{\l@dmodforedtext}%
6650 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
6651 %
```

### XXIX.2.2 Counters, boxes and lengths

`\l@dampcount` `\l@dampcount` is a counter for the & column dividers and `\l@dcolcount` is a counter  
`\l@dcolcount` for the columns.

```
6652 \newcount\l@dampcount
6653 \l@dampcount=1\relax
6654 \newcount\l@dcolcount
6655 \l@dcolcount=0\relax
6656
6657 %
```

`\hilfsbox` Some (temporary) helper items.

```
\hilfsskip
\Hilfsbox
\hilfscount
6658 \newbox\hilfsbox
6659 \newskip\hilfsskip
6660 \newbox\Hilfsbox
6661 \newcount\hilfscount
6662
6663 %
```

30 columns should be adequate (compared to the original 60). These are the column widths. (Originally these were German spelled numbers e.g., `\eins`, `\zwei`, etc).

```
6664 \newdimen\dcoli
6665 \newdimen\dcolii
6666 \newdimen\dcoliii
6667 \newdimen\dcoliv
6668 \newdimen\dcolv
6669 \newdimen\dcolvi
6670 \newdimen\dcolvii
6671 \newdimen\dcolviii
6672 \newdimen\dcolix
6673 \newdimen\dcolx
6674 \newdimen\dcolxi
6675 \newdimen\dcolxii
6676 \newdimen\dcolxiii
6677 \newdimen\dcolxiv
6678 \newdimen\dcolxv
6679 \newdimen\dcolxvi
6680 \newdimen\dcolxvii
6681 \newdimen\dcolxviii
6682 \newdimen\dcolxix
```

```

6683 \newdimen\dcollx
6684 \newdimen\dcollxi
6685 \newdimen\dcollxii
6686 \newdimen\dcollxiii
6687 \newdimen\dcollxiv
6688 \newdimen\dcollxv
6689 \newdimen\dcollxvi
6690 \newdimen\dcollxvii
6691 \newdimen\dcollxviii
6692 \newdimen\dcollxix
6693 \newdimen\dcollxxx
6694 \newdimen\dcollerr % added for error handling
6695
6696 %

```

`\l@dcollwidth` This is a cunning way of storing the columnwidths indexed by the column number `\l@dcollcount`, like an array. (was `\Dimenzuordnung`)

```

6697 \newcommand{\l@dcollwidth}{\ifcase \the\l@dcollcount \dcoli %???
6698 \or \dcolii \or \dcoliii \or \dcoliiii
6699 \or \dcoliv \or \dcolv \or \dcolvi
6700 \or \dcolvii \or \dcolviii \or \dcolix \or \dcolx
6701 \or \dcolxi \or \dcolxii \or \dcolxiii
6702 \or \dcolxiv \or \dcolxv \or \dcolxvi
6703 \or \dcolxvii \or \dcolxviii \or \dcolxix \or \dcolxx
6704 \or \dcolxxi \or \dcolxxii \or \dcolxxiii
6705 \or \dcolxxiv \or \dcolxxv \or \dcolxxvi
6706 \or \dcolxxvii \or \dcolxxviii \or \dcolxxix \or \dcolxxx
6707 \else \dcollerr \fi}
6708
6709 %

```

`\stepl@dcollcount` This increments the column counter, and issues an error message if it is too large.

```

6710 \newcommand*\stepl@dcollcount{\advance\l@dcollcount\@ne
6711 \ifnum\l@dcollcount>30\relax
6712 \led@err@TooManyColumns
6713 \fi}
6714
6715 %

```

`\l@dsetmaxcollwidth` Sets the column width to the maximum value seen so far.

```

6716 \newcommand{\l@dsetmaxcollwidth}{%
6717 \ifdim\l@dcollwidth < \wd\hilfsbox
6718 \l@dcollwidth = \wd\hilfsbox
6719 \else \relax \fi}
6720
6721 %

```



`\measurecell` Measure (recursively) the width required for a math cell.

```

6722 \def\measurecell #1&{&
6723   \ifx #1\ \ifnum\l@dc@lcount=0\let\NEXT\relax%
6724     \else\l@dcheckcols%
6725       \l@dc@lcount=0%
6726       \let\NEXT\measurecell%
6727     \fi%
6728   \else\setbox\hilfsbox=\hbox{&#1&}%
6729     \step\l@dc@lcount%
6730     \l@dsetmaxcolwidth%
6731     \let\NEXT\measurecell%
6732   \fi\NEXT}
6733
6734 %

```

`\measuretcell` Measure (recursively) the width required for a text cell.

```

6735 \def\measuretcell #1&{&
6736   \ifx #1\ \ifnum\l@dc@lcount=0\let\NEXT\relax%
6737     \else\l@dcheckcols%
6738       \l@dc@lcount=0%
6739       \let\NEXT\measuretcell%
6740     \fi%
6741   \else\setbox\hilfsbox=\hbox{&#1&}%
6742     \step\l@dc@lcount%
6743     \l@dsetmaxcolwidth%
6744     \let\NEXT\measuretcell%
6745   \fi\NEXT}
6746
6747 %

```

`\measuremrow` Measure (recursively) the width required for a math row.

```

6748 \def\measuremrow #1\{&
6749   \ifx #1&\let\NEXT\relax%
6750   \else\measurecell #1&\&\&%
6751     \let\NEXT\measuremrow%
6752   \fi\NEXT}
6753 %

```

`\measuretrow` Measure (recursively) the width required for a text row.

```

6754 \def\measuretrow #1\{&
6755   \ifx #1&\let\NEXT\relax%
6756   \else\measuretcell #1&\&\&%
6757     \let\NEXT\measuretrow%
6758   \fi\NEXT}
6759
6760 %

```

`\edtabcolsep` The length `\edtabcolsep` controls the distance between columns.

```
6761 \newskip\edtabcolsep
6762 \global\edtabcolsep=10pt
6763
6764 %
```

`\variab`<sub>65</sub> `\newcommand{\variab}{\relax}`

```
6766
6767 %
```

`\l@dcheckcols` Check that the number of columns is consistent.

```
6768 \newcommand*\l@dcheckcols{%
6769   \ifnum\l@dcolcount=1\relax
6770   \else
6771     \ifnum\l@dampcount=1\relax
6772     \else
6773       \ifnum\l@dcolcount=\l@dampcount\relax
6774       \else
6775         \l@d@err@UnequalColumns
6776         \fi
6777       \fi
6778       \l@dampcount=\l@dcolcount
6779     \fi}
6780
6781 %
```

`\edfilldimen` A length.

```
6782 \newdimen\edfilldimen
6783 \edfilldimen=0pt
6784
6785 %
```

`\c@addcolcount` A counter to hold the number of a column. We use a roman number so that we can grab the column dimension from `\dcol`. We do not use the `\roman`  $\TeX$  command, because some packages, like `babel` can override it in some specific cases (Greek, for example).

`\theadcolcount`

```
6786 \newcounter{addcolcount}
6787 \renewcommand{\theadcolcount}{\romannumeral \c@addcolcount}
6788 %
```

### XXIX.2.3 Tabular typesetting

`\setmcellright` Typeset (recursively) cells of display math right justified.

```

6789 \def\setmcellright #1{\def\edlabel##1{}}%
6790 \let\edindex\nulledindex
6791 \ifx #1\ \ifnum\l@dc@lcount=0%\removelastskip
6792 \let\Next\relax%
6793 \else\l@dc@lcount=0%
6794 \let\Next=\setmcellright%
6795 \fi%
6796 \else%
6797 \disablel@dtabfeet%
6798 \step1@dc@lcount%
6799 \disable@notes%
6800 \setbox\hilfsbox=\hbox{${\displaystyle{#1}}}%
6801 \restore@notes%
6802 \letsforverteilen%
6803 \hskip\hilfsskip${\displaystyle{#1}}%
6804 \hskip\edtabcolsep%
6805 \let\Next=\setmcellright%
6806 \fi\Next}
6807
6808 %

```

`\settcellright` Typeset (recursively) cells of text right justified.

```

6809 \def\settcellright #1{\def\edlabel##1{}}%
6810 \let\edindex\nulledindex
6811 \ifx #1\ \ifnum\l@dc@lcount=0%\removelastskip
6812 \let\Next\relax%
6813 \else\l@dc@lcount=0%
6814 \let\Next=\settcellright%
6815 \fi%
6816 \else%
6817 \disablel@dtabfeet%
6818 \step1@dc@lcount%
6819 \disable@notes%
6820 \setbox\hilfsbox=\hbox{#1}%
6821 \restore@notes%
6822 \letsforverteilen%
6823 \hskip\hilfsskip#1%
6824 \hskip\edtabcolsep%
6825 \let\Next=\settcellright%
6826 \fi\Next}
6827 %

```

`\setmcelleft` Typeset (recursively) cells of display math left justified.

```

6828 \def\setmcelleft #1{\def\edlabel##1{}}%
6829 \let\edindex\nulledindex
6830 \ifx #1\ \ifnum\l@dc@lcount=0 \let\Next\relax%
6831 \else\l@dc@lcount=0%
6832 \let\Next=\setmcelleft%

```

```

6833     \fi%
6834 \else  \disablel@dtabfeet%
6835        \stepl@dcolcount%
6836        \disable@notes%
6837        \setbox\hilfsbox=\hbox{\displaystyle{#1}}%
6838        \restore@notes%
6839        \letsforverteilen%
6840        $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
6841        \let\Next=\setmcellleft%
6842 \fi\Next}
6843
6844 %

```

**\setmcellleft** Typeset (recursively) cells of text left justified.

```

6845 \def\setmcellleft #1&{\def\edlabel##1{}}%
6846     \let\edindex\nulledindex
6847     \ifx #1\ \ifnum\l@dcolcount=0 \let\Next\relax%
6848         \else\l@dcolcount=0%
6849             \let\Next=\setmcellleft%
6850         \fi%
6851 \else  \disablel@dtabfeet%
6852        \stepl@dcolcount%
6853        \disable@notes%
6854        \setbox\hilfsbox=\hbox{#1}%
6855        \restore@notes%
6856        \letsforverteilen%
6857        #1\hskip\hilfsskip\hskip\edtabcolsep%
6858        \let\Next=\setmcellleft%
6859 \fi\Next}
6860 %

```

**\setmcellcenter** Typeset (recursively) cells of display math centered.

```

6861 \def\setmcellcenter #1&{\def\edlabel##1{}}%
6862     \let\edindex\nulledindex
6863     \ifx #1\ \ifnum\l@dcolcount=0\let\Next\relax%
6864         \else\l@dcolcount=0%
6865             \let\Next=\setmcellcenter%
6866         \fi%
6867 \else  \disablel@dtabfeet%
6868        \stepl@dcolcount%
6869        \disable@notes%
6870        \setbox\hilfsbox=\hbox{\displaystyle{#1}}%
6871        \restore@notes%
6872        \letsforverteilen%
6873        \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6874        \hskip\edtabcolsep%
6875        \let\Next=\setmcellcenter%
6876 \fi\Next}

```

```
6877
6878 %
```

**\settcclcenter** Typeset (recursively) cells of text centered.

```
6879 \def\settcclcenter #1{\def\edlabel##1{}}%
6880 \let\edindex\nulledindex
6881 \ifx #1\ \ifnum\l@dcolcount=0 \let\Next\relax%
6882 \else\l@dcolcount=0%
6883 \let\Next=\settcclcenter%
6884 \fi%
6885 \else \disablel@dtabfeet%
6886 \stepl@dcolcount%
6887 \disable@notes%
6888 \setbox\hilfsbox=\hbox{#1}%
6889 \restore@notes%
6890 \letsforverteilen%
6891 \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6892 \hskip\edtabcolsep%
6893 \let\Next=\settcclcenter%
6894 \fi\Next}
6895
6896 %
```

**\NEXT** \let\NEXT=\relax

```
6898
6899 %
```

**\setmrowright** Typeset (recursively) rows of right justified math.

```
6900 \def\setmrowright #1\{\%
6901 \ifx #1& \let\NEXT\relax
6902 \else \centerline{\setmcellright #1&\&\&}
6903 \let\NEXT=\setmrowright
6904 \fi\NEXT}
6905 %
```

**\settroright** Typeset (recursively) rows of right justified text.

```
6906 \def\settroright #1\{\%
6907 \ifx #1& \let\NEXT\relax
6908 \else \centerline{\settcclright #1&\&\&}
6909 \let\NEXT=\settroright
6910 \fi\NEXT}
6911
6912 %
```

**\setmrowleft** Typeset (recursively) rows of left justified math.

```

6913 \def\setmrowleft #1\{\%
6914     \ifx #1&\let\NEXT\relax
6915     \else \centerline{\setmcellleft #1&\&\&}
6916         \let\NEXT=\setmrowleft
6917     \fi\NEXT}
6918 %

```

**\settrorleft** Typeset (recursively) rows of left justified text.

```

6919 \def\settrorleft #1\{\%
6920     \ifx #1& \let\NEXT\relax
6921     \else \centerline{\settrcellleft #1&\&\&}
6922         \let\NEXT=\settrorleft
6923     \fi\NEXT}
6924 %
6925 %

```

**\setmrowcenter** Typeset (recursively) rows of centered math.

```

6926 \def\setmrowcenter #1\{\%
6927     \ifx #1& \let\NEXT\relax%
6928     \else \centerline{\setmcellcenter #1&\&\&}
6929         \let\NEXT=\setmrowcenter
6930     \fi\NEXT}
6931 %

```

**\settrorcenter** Typeset (recursively) rows of centered text.

```

6932 \def\settrorcenter #1\{\%
6933     \ifx #1& \let\NEXT\relax
6934     \else \centerline{\settrcellcenter #1&\&\&}
6935         \let\NEXT=\settrorcenter
6936     \fi\NEXT}
6937 %
6938 %

```

**\nullsetzen**<sup>39</sup> \newcommand{\nullsetzen}{\%

```

6940     \step1@dcolcount%
6941     \l@dcolwidth=0pt%
6942     \ifnum\l@dcolcount=30\let\NEXT\relax%
6943         \l@dcolcount=0\relax
6944     \else\let\NEXT\nullsetzen%
6945     \fi\NEXT}
6946 %
6947 %

```

**\edatleft** \edatleft [*math*]{*symbol*}{*len*}. Left *symbol*, 2*len* high with prepended *math* vertically centered.

```

6948 \newcommand{\edatleft}[3][\@empty]{%
6949   \ifx#1\@empty
6950     \vbox to 10pt{\vss\hbox{\left#2\vrule width0pt height #3
6951       depth 0pt \right. $\hss}\vfil}
6952   \else
6953     \vbox to 4pt{\vss\hbox{\left#2\vrule width0pt height #3
6954       depth 0pt \right. $\vfil}
6955   \fi}
6956 %

```

`\edatright` `\edatright` [*math*]{*symbol*}{*len*}. Right *symbol*, 2*len* high with appended *math* vertically centered.

```

6957 \newcommand{\edatright}[3][\@empty]{%
6958   \ifx#1\@empty
6959     \vbox to 10pt{\vss\hbox{\left.\vrule width0pt height #3
6960       depth 0pt \right#2 $\hss}\vfil}
6961   \else
6962     \vbox to 4pt{\vss\hbox{\left.\vrule width0pt height #3
6963       depth 0pt \right#2 #1 $\vfil}
6964   \fi}
6965 %
6966 %

```

`\edvertline` `\edvertline`{*len*} vertical line *len* high.

```

6967 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
6968 %
6969 %

```

`\edvertdots` `\edvertdots`{*len*} vertical dotted line *len* high.

```

6970 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6971   {\cleaders\hbox{\m@th\hbox{.}\vbox to 0.5em{ }}\vfil}}}
6972 %
6973 %

```

`\l@dtabaddcols` `\l@dtabaddcols`{*startcol*}{*endcol*} adds the widths of the columns *startcol* through *endcol* to `\edfilldimen`. It is a  $\TeX$  style reimplement of the original `\@add@`.

```

6974 \newcommand{\l@dtabaddcols}[2]{%
6975   \l@dcheckstartend{#1}{#2}%
6976   \ifl@dstartendok
6977     \setcounter{addcolcount}{#1}%
6978     \@whilenum \value{addcolcount}<#2\relax \do
6979     {\advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
6980     \advance\edfilldimen by \edtabcolsep
6981     \stepcounter{addcolcount}}%
6982     \advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
6983   \fi

```

```
6984 }
6985
6986 %
```

`\ifl@dstartendok` `\l@dcheckstartend{⟨startcol⟩}{⟨endcol⟩}` checks that the values of `⟨startcol⟩` and `⟨endcol⟩` are sensible. If they are then `\ifl@dstartendok` is set TRUE, otherwise it is set FALSE.

```
6987 \newif\ifl@dstartendok
6988 \newcommand{\l@dcheckstartend}[2]{%
6989   \l@dstartendoktrue
6990   \ifnum #1<\@ne
6991     \l@dstartendokfalse
6992     \led@err@LowStartColumn
6993   \fi
6994   \ifnum #2>30\relax
6995     \l@dstartendokfalse
6996     \led@err@HighEndColumn
6997   \fi
6998   \ifnum #1>#2\relax
6999     \l@dstartendokfalse
7000     \led@err@ReverseColumns
7001   \fi
7002 }
7003
7004 %
```

`\edrowfill` `\edrowfill{⟨startcol⟩}{⟨endcol⟩}` fill fills columns `⟨startcol⟩` to `⟨endcol⟩` inclusive with `⟨fill⟩` (e.g. `\hrulefill`, `\upbracefill`). This is a  $\TeX$  style reimplementation and generalization of the original `\waklam`, `\Waklam`, `\waklamec`, `\wastricht` and `\wapunktetel` macros.

```
7005 \newcommand*\edrowfill}[3]{%
7006   \l@dtabaddcols{#1}{#2}%
7007   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
7008 \let\@edrowfill@=\edrowfill
7009 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
7010
7011 %
```

`\edbeforetab`      The macro `\edbeforetab{⟨text⟩}{⟨math⟩}` puts `⟨text⟩` at the left margin before array cell entry `⟨math⟩`. Conversely, the macro `\edaftertab{⟨math⟩}{⟨text⟩}` puts `⟨text⟩` at the right margin after array cell entry `⟨math⟩`. `\edbeforetab` should be in the first column and `\edaftertab` in the last column. The following macros support these.

`\leftltab`      `\leftltab{⟨text⟩}` for `\edbeforetab` in `\ltab`.

```
7012 \newcommand{\leftltab}[1]{%
7013   \hb@xt@\z@{\vbox{\edtabindent%
```



```

7014 \moveleft\Hilfsskip\hbox{\ #1}\hss}}
7015
7016 %

```

**\lefttab** `\lefttab{<text>}{<math>}` for `\edbeforetab` in `\rtab`.

```

7017 \newcommand{\lefttab}[2]{%
7018 #2\hb@xt@z@{\vbox{\edtabindent%
7019 \advance\Hilfsskip by\dcoli%
7020 \moveleft\Hilfsskip\hbox{\ #1}\hss}}
7021
7022 %

```

**\leftctab** `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`.

```

7023 \newcommand{\leftctab}[2]{%
7024 \hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
7025 \advance\Hilfsskip by 0.5\dcoli%
7026 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7027 \disablel@dtabfeet$\displaystyle{#2}$}%
7028 \advance\Hilfsskip by -0.5\wd\hilfsbox%
7029 \moveleft\Hilfsskip\hbox{\ #1}\hss}}%
7030 #2}
7031
7032 %

```

**\rightctab** `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`.

```

7033 \newcommand{\rightctab}[2]{%
7034 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7035 \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
7036 #1\hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
7037 \advance\Hilfsskip by 0.5\l@dcolwidth%
7038 \advance\Hilfsskip by -\wd\hilfsbox%
7039 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7040 \disablel@dtabfeet$\displaystyle{#1}$}%
7041 \advance\Hilfsskip by -0.5\wd\hilfsbox%
7042 \advance\Hilfsskip by \edtabcolsep%
7043 \moveright\Hilfsskip\hbox{ #2}\hss}}%
7044 }
7045
7046 %

```

**\rightltab** `\rightltab{<math>}{<text>}` for `\edaftertab` in `\ltab`.

```

7047 \newcommand{\rightltab}[2]{%
7048 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7049 \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
7050 #1\hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
7051 \advance\Hilfsskip by\l@dcolwidth%

```

```

7052 \advance\Hilfsskip by-\wd\hilfsbox%
7053 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7054 \disablel@dtabfeet$\displaystyle{#1}$}%
7055 \advance\Hilfsskip by-\wd\hilfsbox%
7056 \advance\Hilfsskip by\edtabcolsep%
7057 \moveright\Hilfsskip\hbox{ #2}}\hss}%
7058 }
7059
7060 %

```

**\righttab** `\righttab{math}{text}` for `\edaftertab` in `\rtab`.

```

7061 \newcommand{\righttab}[2]{%
7062 \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
7063 \disablel@dtabfeet#2}%
7064 #1\hb@xt@\z@\vbox{\edtabindent%
7065 \advance\Hilfsskip by-\wd\hilfsbox%
7066 \advance\Hilfsskip by\edtabcolsep%
7067 \moveright\Hilfsskip\hbox{ #2}}\hss}%
7068 }
7069
7070 %

```

**\rtab** `\rtab{body}` typesets *body* as an array with the entries right justified.

**\edbeforetab** The process is first to measure the *body* to get the column widths, and then in a  
**\edaftertab** second pass to typeset the body.

```

7071 \newcommand{\rtab}[1]{%
7072 \l@dnnullfills
7073 \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
7074 \def\edaftertab##1##2{\righttab{##1}{##2}}%
7075 \measurebody{#1}%
7076 \l@drestorefills
7077 \variab
7078 \setmrowright #1\&\%
7079 \enablel@dtabfeet}
7080
7081 %

```

**\measurebody** `\measurebody{body}` measures the array *body*.

```

7082 \newcommand{\measurebody}[1]{%
7083 \disablel@dtabfeet%
7084 \l@dcolcount=0%
7085 \nullsetzen%
7086 \l@dcolcount=0
7087 \measuremrow #1\&\%
7088 \global\l@dampcount=1}
7089
7090 %

```

`\rtabtext` `\rtabtext{⟨body⟩}` typesets *⟨body⟩* as a tabular with the entries right justified.

```

7091 \newcommand{\rtabtext}[1]{%
7092   \l@dnnullfills
7093   \measuretbody{#1}%
7094   \l@drestorefills
7095   \variab
7096   \settroright #1\&\%
7097   \enablel@dtabfeet}
7098
7099 %

```

`\measuretbody` `\measuretbody{⟨body⟩}` measures the tabular *⟨body⟩*.

```

7100 \newcommand{\measuretbody}[1]{%
7101   \disable@notes%
7102   \disablel@dtabfeet%
7103   \l@dcolcount=0%
7104   \nullsetzen%
7105   \l@dcolcount=0
7106   \measuretrorow #1\&\%
7107   \restore@notes%
7108   \global\l@dampcount=1}
7109
7110 %

```

`\ltab` Array with entries left justified.

```

\edbeforetab
\edaftertab
7111 \newcommand{\ltab}[1]{%
7112   \l@dnnullfills
7113   \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
7114   \def\edaftertab##1##2{\rightltab{##1}{##2}}%
7115   \measuretbody{#1}%
7116   \l@drestorefills
7117   \variab
7118   \settrorowleft #1\&\%
7119   \enablel@dtabfeet}
7120
7121 %

```

`\ltabtext` Tabular with entries left justified.

```

7122 \newcommand{\ltabtext}[1]{%
7123   \l@dnnullfills
7124   \measuretbody{#1}%
7125   \l@drestorefills
7126   \variab
7127   \settrorowleft #1\&\%
7128   \enablel@dtabfeet}
7129
7130 %

```

`\ctab` Array with centered entries.

```

\edbeforetab
\edaftertab
7131 \newcommand{\ctab}[1]{%
7132   \l@dnnullfills
7133   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
7134   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
7135   \measurebody{#1}%
7136   \l@drestorefills
7137   \variab
7138   \setmrowcenter #1\&\&%
7139   \enablel@dtabfeet}
7140
7141 %

```

`\ctabtext` Tabular with entries centered.

```

7142 \newcommand{\ctabtext}[1]{%
7143   \l@dnnullfills
7144   \measuretbody{#1}%
7145   \l@drestorefills
7146   \variab
7147   \settrrowcenter #1\&\&%
7148   \enablel@dtabfeet}
7149
7150 %

```

```

\spreadtext51 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
7152   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
7153 %

```

```

\spreadmath54 \newcommand{\spreadmath}[1]{%
7155   \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
7156
7157 %

```

`\HILFSskip` More helpers.

```

\Hilfsskip
7158 \newskip\HILFSskip
7159 \newskip\Hilfsskip
7160
7161 %

```

```

\EDTABINDENT62 \newcommand{\EDTABINDENT}{%
7163   \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
7164   \else\step\l@dcolcount%
7165   \advance\Hilfsskip by\l@dcolwidth%
7166   \ifdim\l@dcolwidth=0pt\advance\hilfscout\@ne

```

```

7167     \else\advance\Hilfsskip by \the\hilfscout\edtabcolsep%
7168     \hilfscout=1\fi%
7169     \let\NEXT=\EDTABINDENT%
7170 \fi\NEXT}%
7171 %

```

`\edtabindent` (was `\tabindent`)

```

7172 \newcommand{\edtabindent}{%
7173   \l@dcolcount=0\relax
7174   \Hilfsskip=0pt%
7175   \hilfscout=1\relax
7176   \EDTABINDENT%
7177   \hilfsskip=\hsize%
7178   \advance\hilfsskip -\Hilfsskip%
7179   \Hilfsskip=0.5\hilfsskip%
7180 }%
7181 %
7182 %

```

`\EDTAB` (was `\TAB`)

```

7183 \def\EDTAB #1|#2|{%
7184   \setbox\tabhilfbox=\hbox{\displaystyle{#1}}%
7185   \setbox\tabHilfbox=\hbox{\displaystyle{#2}}%
7186   \advance\tabelskip -\wd\tabhilfbox%
7187   \advance\tabelskip -\wd\tabHilfbox%
7188   \unhbox\tabhilfbox\hskip\tabelskip%
7189   \unhbox\tabHilfbox}%
7190 %
7191 %

```

`\EDTABtext` (was `\TABtext`)

```

7192 \def\EDTABtext #1|#2|{%
7193   \setbox\tabhilfbox=\hbox{#1}%
7194   \setbox\tabHilfbox=\hbox{#2}%
7195   \advance\tabelskip -\wd\tabhilfbox%
7196   \advance\tabelskip -\wd\tabHilfbox%
7197   \unhbox\tabhilfbox\hskip\tabelskip%
7198   \unhbox\tabHilfbox}%
7199 %

```

`\tabhilfbox` Further helpers.

```

\newbox\tabhilfbox
\newbox\tabHilfbox
7200
7201
7202
7203 %

```

**XXIX.2.4 Environments**

`edarrayl edarrayc edarrayr` The ‘environment’ forms for `\ltab`, `\ctab` and `\rtab`.

```
7204 \newenvironment{edarrayl}{\l@dcollect@body\ltab}{\}
7205 \newenvironment{edarrayc}{\l@dcollect@body\ctab}{\}
7206 \newenvironment{edarrayr}{\l@dcollect@body\rtab}{\}
7207
7208 %
```

`edtabularl edtabularc edtabularr` The ‘environment’ forms for `\ltabtext`, `\ctabtext` and `\rtabtext`.

```
7209 \newenvironment{edtabularl}{\l@dcollect@body\ltabtext}{\}
7210 \newenvironment{edtabularc}{\l@dcollect@body\ctabtext}{\}
7211 \newenvironment{edtabularr}{\l@dcollect@body\rtabtext}{\}
7212
7213 %
```

**XXX Quotation's commands**

`\initnumbering@quote` This macro, called at the beginning of any numbered section, locally redefines the quotation and quote environments, in order to allow their use inside of numbered sections.

```
\quotation \initnumbering@quote defines quotation environment.
\endquotation
\quote
\endquote
7214 \newcommand{\initnumbering@quote}{
7215   \ifnoquotation@else
7216   \renewcommand{\quotation}{\par\leavevmode%
7217     \parindent=1.5em%
7218     \skipnumbering%
7219     \ifautopar%
7220       \vskip-\parskip%
7221     \else%
7222       \vskip\topsep%
7223     \fi%
7224     \global\leftskip=\leftmargin%
7225     \global\rightskip=\leftmargin%
7226   }
7227   \renewcommand{\endquotation}{\par%
7228     \global\leftskip=0pt%
7229     \global\rightskip=0pt%
7230     \leavevmode%
7231     \skipnumbering%
7232     \ifautopar%
7233       \vskip-\parskip%
7234     \else%
7235       \vskip\topsep%
7236     \fi%
```

```

7237     }
7238     \renewcommand{\quote}{\par\leavevmode%
7239         \parindent=0pt%
7240         \skipnumbering%
7241         \ifautopar%
7242             \vskip-\parskip%
7243         \else%
7244             \vskip\topsep%
7245         \fi%
7246         \global\leftskip=\leftmargin%
7247         \global\rightskip=\leftmargin%
7248     }
7249     \renewcommand{\endquote}{\par%
7250         \global\leftskip=0pt%
7251         \global\rightskip=0pt%
7252         \leavevmode%
7253         \skipnumbering%
7254         \ifautopar%
7255             \vskip-\parskip%
7256         \else%
7257             \vskip\topsep%
7258         \fi%
7259     }
7260 \fi
7261 }
7262 %

```

## XXXI Section's title commands

### XXXI.1 Commands to disable some feature

`\ledsectnotoc` The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```

7263 \newcommand{\ledsectnotoc}{\let\addcontentsline@gobblethree}
7264 %

```

`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```

7265 \newcommand{\ledsectnomark}{%
7266     \let\chaptermark@gobble%
7267     \let\sectionmark@gobble%
7268     \let\subsectionmark@gobble%
7269 }
7270 %

```

### XXXI.2 General overview

The system of `\eledxxxx` commands to section text work like this:

1. When one of these commands is called, `reledmac` writes to an auxiliary files:
  - The section level.
  - The section title.
  - The side (when `eledpar` is used).
  - The `pstart` where the command is called.
  - If we have starred version or not.
2. `reledmac` adds the title of the section to `pstart`, as normal content. This is to enable critical notes.
3. When  $\LaTeX$  is run a other time, this file is read. That:
  - Adds the `pstart` number to a list of `pstarts` where a sectioning command is used.
  - Defines a command, the name of which contains the `pstart` number, and which calls the normal  $\LaTeX$  sectioning command.
4. This last command is called when the `pstart` is effectively printed.

### XXXI.3 `\beforeeledchapter` command

We do not define commands for `\eledsection` and related if the `noeledsec` option is loaded. We use `etoolbox` tests and not the `\ifxxx...\else...\fi` structure to prevent problem of expansions with command after the `\ifxxx` which contains `\fi`. As we patch command inside this test, we need to change the category code of `#` character *before* `\notbool` statement, because the second argument is read with the standard `catcode` (read *The TeXbook* to understand when the `catcode`'s change has effect).

```
7271 \catcode\#=12
7272 \notbool{@noeled@sec}{%
7273 %
```

`\beforeeledchapter` For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by `eledmac`. Users have to use `\beforeeledchapter`.

```
7274 \ifl@dmemoir
7275   \newcommand\beforeeledchapter{%
7276     \clearforchapter%
7277   }
7278 \else
7279   \newcommand\beforeeledchapter{%
7280     \if@openright%
7281       \cleardoublepage%
7282     \else%
7283       \clearpage%
7284     \fi%
7285   }
7286 \fi
7287 %
```



### XXXI.4 Auxiliary commands

`\if@eled@sectioning` The boolean `\if@eled@sectioning` is set to true when a sectioning command is called by a `\eledxxx` command, and set to false after. It is used to enable/disable line number printing.

```
7288 \newif\if@eled@sectioning
7289 %
```

`\print@leftmargin@eledsection` and `\print@rightmargin@eledsection` are added by `reledmac` inside the code of sectioning command, in order to affix lines numbers. They include tests for RTL languages.

```
7290 \def\print@rightmargin@eledsection{%
7291   \if@eled@sectioning%
7292   \begingroup%
7293   \if@RTL%
7294     \let\llap\rlap%
7295     \let\leftlinenum\rightlinenum%
7296     \let\leftlinenumR\rightlinenumR%
7297     \let\l@drd@ta\l@dld@ta%
7298     \let\l@dlsn@te\l@dlsn@te%
7299   \fi%
7300   \hfill\l@drd@ta \csuse{LR}{\l@dlsn@te}%
7301   \endgroup%
7302 \fi%
7303 }%
7304
7305 \def\print@leftmargin@eledsection{%
7306   \if@eled@sectioning%
7307   \leavevmode%
7308   \begingroup%
7309   \if@RTL%
7310     \let\rlap\llap%
7311     \let\rightlinenum\leftlinenum%
7312     \let\rightlinenumR\leftlinenumR%
7313     \let\l@dld@ta\l@drd@ta%
7314     \let\l@dlsn@te\l@dlsn@te%
7315   \fi%
7316   \l@dld@ta\csuse{LR}{\l@dlsn@te}%
7317   \endgroup%
7318 \fi%
7319 }%
7320
7321 %
```

### XXXI.5 Patching standard commands

`\chapter` We have to patch  $\LaTeX$ , `book` and `memoir` sectioning commands in order to:

```
\M@sect
\@mem@old@ssect
\@makechapterhead
\@makechapterhead
\@makeschapterhead
\@sect
\@ssect
```

- Disable `\edtext` inside.
- Disable page breaking (for `\chapter`).
- Add line numbers and sidenotes.

Unfortunately, Maïeul Rouquette was not able to try if `memoir` is loaded. That is why `eledmac` tries to define for both standard class and `memoir` class.

```

7322 \AtBeginDocument{%
7323 \patchcmd{\chapter}{\clearforchapter}{%
7324   \if@eled@sectioning\else%
7325   \ifl@dprintingpages\else%
7326   \clearforchapter%
7327   \fi%
7328   \fi%
7329 }
7330 {}
7331 {}
7332
7333
7334 \pretocmd{\M@sect}
7335   {\let\old@edtext=\edtext%
7336   \let\edtext=\dummy@edtext@showlemma%
7337   }
7338   {}
7339   {}
7340
7341 \apptocmd{\M@sect}
7342   {\let\edtext=\old@edtext}
7343   {}
7344   {}
7345
7346 \patchcmd{\M@sect}
7347   { #9}
7348   { #9%
7349   \print@rightmargin@eledsection%
7350   }
7351   {}
7352   {}
7353
7354 \patchcmd{\M@sect}
7355   {\hskip #3\relax}
7356   {\hskip #3\relax%
7357   \print@leftmargin@eledsection%
7358   }
7359   {}
7360   {}
7361
7362 \patchcmd{\@mem@old@ssect}
7363   {#5}

```

```

7364   {#5%
7365   \print@leftmargin@eledsection%
7366   }
7367   {}
7368   {}
7369
7370 \patchcmd{\@mem@old@ssect}
7371   {\hskip #1}
7372   {\hskip #1%
7373   \print@rightmargin@eledsection%
7374   }
7375   {}
7376   {}
7377
7378 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
7379   \if@eled@sectioning\else%
7380   \ifl@dprintingpages\else%
7381   \if@openright\cleardoublepage\else\clearpage\fi}No clearpage inside a
\Pages: will keep critical notes from printing on the title page. Here for
classical classes
7382   \fi%
7383   \fi%
7384   }%
7385   {}%
7386   {}%
7387
7388 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\
fi}{%
7389   \if@eled@sectioning\else%
7390   \ifl@dprintingpages\else%
7391   \if@openright\cleardoublepage\else\clearpage\fi}No clearpage inside a
\Pages: will keep critical notes from printing on the title page. Here for
scrbook.
7392   \fi%
7393   \fi%
7394   }
7395   {}
7396   {}
7397
7398 \patchcmd{\@makechapterhead}
7399   {#1}
7400   {\print@leftmargin@eledsection%
7401   #1%
7402   \print@rightmargin@eledsection%
7403   }
7404   {}
7405   {}
7406
7407 \patchcmd{\@makechapterhead}% For BIDI
7408   {\if@RTL\raggedleft\else\raggedright\fi}%

```

```

7409 {\if@eled@sectioning\else%
7410 \if@RTL\raggedleft\else\raggedright\fi%
7411 \fi%
7412 }%
7413 {}%
7414 {}%
7415
7416 \patchcmd{\@makeschapterhead}
7417   {#1}
7418   {\print@leftmargin@eledsection%
7419     #1%
7420     \print@rightmargin@eledsection%
7421   }
7422   {}
7423   {}
7424
7425 \pretocmd{\@sect}
7426   {\let\old@edtext=\edtext
7427     \let\edtext=\dummy@edtext@showlemma%
7428   }
7429   {}
7430   {}
7431
7432 \apptocmd{\@sect}
7433   {\let\edtext=\old@edtext}
7434   {}
7435   {}
7436
7437 \pretocmd{\@ssect}
7438   {\let\old@edtext=\edtext%
7439     \let\edtext=\dummy@edtext@showlemma%
7440   }
7441   {}
7442   {}
7443
7444 \apptocmd{\@ssect}
7445   {\let\edtext=\old@edtext}
7446   {}
7447   {}
7448
7449 %

```

hyperref also redefines \@sect. That is why, when manipulating arguments, we patch \@sect and the same only if hyperref is not used. If it is, we patch the \NR commands.

```

7450 \@ifpackageloaded{nameref}{
7451
7452   \patchcmd{\NR@sect}
7453     {#8}
7454     {#8%
7455       \print@rightmargin@eledsection%

```

```

7456     }
7457     {}
7458     {}
7459
7460 \patchcmd{\NR@sect}
7461   {\hskip #3\relax}
7462   {\hskip #3\relax%
7463   \print@leftmargin@eledsection%
7464   }
7465   {}
7466   {}
7467
7468 \patchcmd{\NR@ssect}
7469   {#5}
7470   {#5%
7471   \print@rightmargin@eledsection%
7472   }
7473   {}
7474   {}
7475
7476 \patchcmd{\NR@ssect}
7477   {\hskip #1}
7478   {\hskip #1%
7479   \print@leftmargin@eledsection%
7480   }
7481   {}
7482   {}
7483 }%
7484 {
7485 \patchcmd{\@sect}
7486   {#8}
7487   {#8%
7488   \print@rightmargin@eledsection%
7489   }
7490   {}
7491   {}
7492
7493 \patchcmd{\@sect}
7494   {\hskip #3\relax}
7495   {\hskip #3\relax%
7496   \print@leftmargin@eledsection%
7497   }
7498   {}
7499   {}
7500
7501 \patchcmd{\@ssect}
7502   {#5}
7503   {#5%
7504   \print@rightmargin@eledsection%
7505   }

```

```

7506     {}
7507     {}
7508
7509     \patchcmd{\@ssect}
7510         {\hskip #1}
7511         {\hskip #1%
7512         \print@leftmargin@eledsection%
7513         }
7514     {}
7515     {}
7516 }%
7517 }
7518 %

```

Now, we have finished to patch the commands, using # with a catcode equals to 12. We close the `\notbool{@noeled@sec}` statement, restore the normal catcode for # and reopen a new `\notbool{@noeled@sec}` statement.

```

7519 {}}%
7520 \protect\catcode`\#=6 %Space NEEDS by \catcode
7521 \notbool{@noeled@sec}{%
7522 %

```

### XXXI.6 Main code of `\eledxxx` commands

`\eled@sectioning@out` `\eled@sectioning@out` is the output file, to dump the pstarts where a sectioning command is used.

```

7523 \newwrite\eled@sectioning@out
7524 %

```

`\eledchapter` `\eledsection` And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

```

7525 \eledsubsection \newcommand{\eledchapter}[2] [] {%
7526 \eledsubsubsection #2%
7527 \eledchapter* \ifledRcol%
7528 \eledsection* \immediate\write\eled@sectioningR@out{%
7529 \eledsubsection* \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
7530 \eledsubsubsection* }%
7531 \else%
7532 \immediate\write\eled@sectioning@out{%
7533 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
7534 }%
7535 \fi%
7536 }
7537
7538 \newcommand{\eledsection}[2] [] {%
7539 #2%
7540 \ifledRcol%

```

```

7541 \immediate\write\eled@sectioningR@out{%
7542 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
7543 }%
7544 \else%
7545 \immediate\write\eled@sectioning@out{%
7546 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
7547 }%
7548 \fi%
7549 }
7550
7551 \newcommand{\eledsubsection}[2][R]{%
7552 #2%
7553 \ifledRcol%
7554 \immediate\write\eled@sectioningR@out{%
7555 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
7556 }%
7557 \else%
7558 \immediate\write\eled@sectioning@out{%
7559 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
7560 }%
7561 \fi%
7562 }
7563 \newcommand{\eledsubsubsection}[2][R]{%
7564 #2%
7565 \ifledRcol%
7566 \immediate\write\eled@sectioningR@out{%
7567 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
7568 }{R}
7569 }%
7570 \else%
7571 \immediate\write\eled@sectioning@out{%
7572 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}
7573 }{L}
7574 }%
7575 \fi%
7576 }
7577 \WithSuffix\newcommand\eledchapter*[2][R]{%
7578 #2%
7579 \ifledRcol%
7580 \immediate\write\eled@sectioningR@out{%
7581 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
7582 }%
7583 \else%
7584 \immediate\write\eled@sectioning@out{%
7585 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
7586 }%
7587 \fi%

```

```

7588 }
7589
7590 \WithSuffix\newcommand\eledsection*[2] [] {%
7591   #2%
7592   \ifledRcol%
7593     \immediate\write\eled@sectioningR@out{%
7594       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
7595     }%
7596   \else%
7597     \immediate\write\eled@sectioning@out{%
7598       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
7599     }%
7600   \fi%
7601 }
7602
7603 \WithSuffix\newcommand\eledsubsection*[2] [] {%
7604   #2%
7605   \ifledRcol%
7606     \immediate\write\eled@sectioningR@out{%
7607       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
7608     }%
7609   \else%
7610     \immediate\write\eled@sectioning@out{%
7611       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}
7612     }{*}{L}
7613   \fi%
7614 }
7615
7616 \WithSuffix\newcommand\eledsubsubsection*[2] [] {%
7617   #2%
7618   \ifledRcol%
7619     \immediate\write\eled@sectioningR@out{%
7620       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
7621     }{*}{R}
7622   \else%
7623     \immediate\write\eled@sectioning@out{%
7624       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}
7625     }{*}{L}
7626   \fi%
7627 }
7628 %

```

### XXXI.7 Macros written in the auxiliary file

`\eled@chapter` The sectioning macros, called in the auxiliary file. They have five arguments:  
`\eled@section`  
`\eled@subsection`  
`\eled@subsubsection`



1. Optional arguments of  $\LaTeX$  sectioning command.
2. Mandatory arguments of  $\LaTeX$  sectioning command.
3. Pstart number.
4. Side: R if right, nothing if left.
5. Starred or not.

```

7629 \def\eled@chapter#1#2#3#4#5{%
7630   \ifstrempy{#4}%
7631   {%
7632     \ifstrempy{#1}%
7633     {%
7634       \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%
7635       \global\csdef{eled@ssectmark@#3#5}{\let\edtext=\dummy@edtext{\
chaptermark{#2}}}%
7636       }%Need for \pairs, because of using parbox.
7637       {%
7638         \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter[#1]{#2}}%
7639         \global\csdef{eled@ssectmark@#3#5}{\let\edtext=\dummy@edtext{\
chaptermark{#2}}}%Need for \pairs, because of using parbox.
7640         }%
7641       }%
7642     }%
7643     \ifstrempy{#1}%
7644     {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%
7645     {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
7646     }%
7647     \listcsadd{eled@sections#5@0}{#3}%
7648   }
7649 \def\eled@section#1#2#3#4#5{%
7650   \ifstrempy{#4}%
7651   {\ifstrempy{#1}%
7652   {%
7653     \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
7654     \global\csdef{eled@ssectmark@#3#5}{\let\edtext=\dummy@edtext{\
sectionmark{#2}}}%Need for \pairs, because of using parbox.
7655     }%
7656     {%
7657       \global\csdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
7658       \global\csdef{eled@ssectmark@#3#5}{\let\edtext=\dummy@edtext{\
sectionmark{#1}}}%Need for \pairs, because of using parbox.
7659     }%
7660   }%

```

```

7661     {\ifstrempy{#1}%
7662      {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
7663      {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
7664     }
7665     \listcsgadd{eled@sections#5@@}{#3}%
7666     }
7667 \def\eled@subsection#1#2#3#4#5{%
7668     \ifstrempy{#4}%
7669     {\ifstrempy{#1}%
7670      {%
7671       \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
7672       \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\csuse
{subsectionmark}{#2}}%Need for \pairs, because of using parbox. \csuse in
case of \subsectionmark is not defined (book)
7673      }%
7674      {%
7675       \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
7676       \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\csuse
{subsectionmark}{#1}}%Need for \pairs, because of using parbox. \csuse in
case of \subsectionmark is not defined (book)
7677      }%
7678     }%
7679     {\ifstrempy{#1}%
7680      {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
7681      {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
7682     }
7683     \listcsgadd{eled@sections#5@@}{#3}%
7684     }
7685 \def\eled@subsubsection#1#2#3#4#5{%
7686     \ifstrempy{#4}%
7687     {\ifstrempy{#1}%
7688      {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
7689      {\global\csdef{eled@sectioning@#3#5}{\subsubsection[#1]{#2}}}%
7690     }%
7691     {\ifstrempy{#1}%
7692      {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
7693      {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
7694     }
7695     \listcsgadd{eled@sections#5@@}{#3}%
7696     }
7697
7698 %

```

End of the conditional test about noeledsec option.

```

7699 }{}
7700 %

```

## XXXII Page breaking or no page breaking depending of specific lines

By default, page breaks are automatic. However, the user can define lines which will force page breaks, or prevent page breaks around one specific line. On the first run, the line-list file records the line number of where the page break is being changed (either forced, or prevented). On the next run, page breaks occur either before or after this line, depending on how the user sets the command. The default setting is after the line.

`\normal@page@break` `\normal@page@break` is an etoolbox list which contains the absolute line number of the last line, for each page.

```
7701 \def\normal@page@break{}
7702 %
```

`\prev@pb` The `\l@prev@pb` macro is a etoolbox list, which contains the lines in which page breaks occur (before or after). The `\l@prev@nopb` macro is a etoolbox list, which contains the lines with NO page break before or after.

```
7703 \def\l@prev@pb{}
7704 \def\l@prev@nopb{}
7705 %
```

`\ledpb` The `\ledpb` macro writes the call to `\led@pb` in line-list file. The `\ledpbnum` macro writes the call to `\led@pbnum` in line-list file. The `\lednopb` macro writes the call to `\led@nopb` in line-list file. The `\lednopbnum` macro writes the call to `\led@nopbnum` in line-list file.

```
7706 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
7707 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
7708 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
7709 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
7710 %
```

`\led@pb` The `\led@pb` adds the absolute line number in the `\prev@pb` list. The `\led@pbnum` adds the argument in the `\prev@pb` list. The `\led@nopb` adds the absolute line number in the `\prev@nopb` list. The `\led@nopbnum` adds the argument in the `\prev@nopb` list.

```
\led@nopbnum
7711 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
7712 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
7713 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
7714 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
7715 %
```

`\ledpbsetting` The `\ledpbsetting` macro only changes the value of `\led@pb@macro`, for which the default value is before.

```
7716 \def\led@pb@setting{before}
7717 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
7718 %
```

`\led@check@pb` and `\led@check@nopb` are called before or after each line. They check if a page break must occur, depending on the current line and on the content of `\l@pb`.

```

7719 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\
pagebreak[4]}{}}
7720 \newcommand{\led@check@nopb}{%
7721   \IfStrEq{\led@pb@setting}{before}{%
7722     \xifinlist{\the\absline@num}{\l@prev@nopb}%
7723     {\numdef{\abs@prevline}{\the\absline@num-1}%
7724     \xifinlist{\abs@prevline}{\normal@page@break}%
7725     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7726     {}}%
7727   {}}%
7728   {}%
7729   {}%
7730 \IfStrEq{\led@pb@setting}{after}{%
7731   \xifinlist{\the\absline@num}{\l@prev@nopb}{%
7732   \xifinlist{\the\absline@num}{\normal@page@break}%
7733   {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7734   {}%
7735 }%
7736   {}%
7737   {}%
7738   {}%
7739 }
7740 %

```

### XXXIII Long verse: prevents being separated by a page break

`\iflednopbinverse` The `\lednopbinverse` boolean is set to false by default. If set to true, `reledmac` will automatically prevent page breaks inside verse. The declaration is made at the beginning of the file, because it is used as a package option.

`\check@pb@in@verse` The `\check@pb@in@verse` checks if a verse is broken in two page. If true, it adds:

- The absolute line number of the first line of the verse -1 in the `\led@pb` list, if the page break must occur before the verse.
- The absolute line number of the first line of the verse -1 in the `\led@nopb` list, if the page break must occur after the verse.

```

7741 \newcommand{\check@pb@in@verse}{%
7742   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
enabling page breaks in verse control, while on a hanging verse.
7743   \ifnum\page@num=\last@page@num\else%If we have change page
7744   \IfStrEq{\led@pb@setting}{before}{%

```

```

7745         \numgdef{\abs@line@verse}{\the\absline@num-1}%
7746         \ledpbnnum{\abs@line@verse}%
7747     }{}%
7748     \IfStrEq{\led@pb@setting}{after}{%
7749         \numgdef{\abs@line@verse}{\the\absline@num-1}%
7750         \lednopbnnum{\abs@line@verse}%
7751     }{}%
7752     \fi%
7753 \fi\fi\fi%
7754 }
7755 %

```

## XXXIV Tools for hyperref package

`\Hy@raisedlink@left` The hyperref package provides a `\Hy@raisedlink` command, to be used to add an anchor to the top of a line and not to the bottom of it.<sup>33</sup>

However, this command disrupts the line breaking mechanism when it is called before any word. This is why `reledmac` defines `\Hy@raisedlink@left` that is called to the left of words, at the beginning of `\edtext` or inside the `\edlabel` commands.<sup>34</sup>

```

7756 \def\Hy@raisedlink@left#1{%
7757     \ifvmode
7758         #1%
7759     \else
7760         \Hy@SaveSpaceFactor
7761         \llap{\smash{%
7762             \begingroup
7763                 \let\HyperRaiseLinkLength\@tempdima
7764                 \setlength\HyperRaiseLinkLength\HyperRaiseLinkDefault
7765                 \HyperRaiseLinkHook
7766             \expandafter\endgroup
7767             \expandafter\raise\the\HyperRaiseLinkLength\hbox{%
7768                 \Hy@RestoreSpaceFactor
7769                 #1%
7770                 \Hy@SaveSpaceFactor
7771             }}%
7772         \Hy@RestoreSpaceFactor
7773         \penalty\@M\hskip\z@
7774     \fi
7775 }
7776 %
7777

```

<sup>33</sup><http://tex.stackexchange.com/a/17138/7712>.

<sup>34</sup>The code is inspired by an answer given by @unbonpetit. Thanks to him. <http://texnique.fr/80/osqa/questions/781/hyraisedlink-perturbe-la-maniere-dont-se-fait-la-coupe-de-ligne/801>.

## XXXV Compatibility with eledmac

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

```

7778 \ifeledmaccompat@%
7779
7780 \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
7781 \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
7782 \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
7783 \newcommand{\footthreecolX}[1]{\XarrangementX[#1]{threecol}}%
7784
7785 \unless\ifnocritical@
7786 \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
7787 \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
7788 \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
7789 \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%
7790 \let\hsizetwocol\Xhsizetwocol
7791 \let\hsizethreecol\Xhsizethreecol
7792 \let\bhookXnote\Xbhooknote
7793 \let\boxsymlinenum\Xboxsymlinenum
7794 \let\symlinenum\Xsymlinenum
7795 \let\beforenumberinfootnote\Xbeforenumber
7796 \let\afternumberinfootnote\Xafternumber
7797 \let\beforeXsymlinenum\Xbeforesymlinenum
7798 \let\afterXsymlinenum\Xaftersymlinenum
7799 \let\inplaceofnumber\Xinplaceofnumber
7800 \let\Xlemmaseparator\lemmaseparator
7801 \let\afterlemmaseparator\Xafterlemmaseparator
7802 \let\beforelemmaseparator\Xbeforelemmaseparator
7803 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
7804 \let\txtbeforeXnotes\Xtxtbeforenotes
7805 \let\afterXrule\Xafterrule
7806 \let\numberonlyfirstinline\Xnumberonlyfirstinline
7807 \let\numberonlyfirstintwolines\Xnumberonlyfirstintwolines
7808 \let\nonumberinfootnote\Xnonumberinfootnote
7809 \let\pstartinfootnote\Xpstart
7810 \let\pstartinfootnoteeverytime\Xpstarteverytime
7811 \let\onlyXpstart\Xonlypstart
7812 \let\Xnonumberinfootnote\Xnonumber
7813 \let\nonbreakableafternumber\Xnonbreakableafternumber
7814 \let\maxhXnotes\Xmaxhnotes
7815 \let\beforeXnotes\Xbeforenotes
7816 \let\boxlinenum\Xboxlinenum
7817 \let\boxlinenumalign\Xboxlinenumalign
7818 \let\boxstartlinenum\Xboxstartlinenum
7819 \let\boxendlinenum\Xboxendlinenum
7820 \let\twolines\Xtwolines
7821 \let\morethantwolines\Xmorethantwolines
7822 \let\twolinesbutnotmore\Xtwolinesbutnotmore
7823 \let\twolinesonlyinsamepage\Xtwolinesonlyinsamepage

```

```

7824 \fi
7825
7826 \unless\ifnofamiliar@
7827   \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
7828 \fi
7829 \newcommandx{\parafootsep}[2][1,usedefault]{%
7830   \Xparafootsep[#1]{#2}%
7831   \parafootsepX[#1]{#2}
7832 }%
7833
7834 \newcommandx{\afternote}[2][1,usedefault]{%
7835   \Xafternote[#1]{#2}%
7836   \afternoteX[#1]{#2}%
7837 }%
7838
7839 \unless\ifnoend@
7840   \let\XendXtwolines\Xendtwolines
7841   \let\XendXmorethantwolines\Xendmorethantwolines
7842   \let\bhookXendnote\Xendbhooknote
7843   \let\boxXendlinenum\Xendboxlinenum%
7844   \let\boxXendlinenumalign\Xendboxlinenumalign%
7845   \let\boxXendstartlinenum\Xendboxstartlinenum%
7846   \let\boxXendendlinenum\Xendboxendlinenum%
7847   \let\XendXlemmaseparator\Xendlemmaseparator
7848   \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
7849   \let\XendXafterlemmaseparator\Xendafterlemmaseparator
7850   \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
7851 \fi
7852
7853 \AtBeginDocument{%
7854   \ifdef\lineref{}\let\lineref\edlineref}%
7855 }%
7856
7857 \fi%
7858 %
7859 %

```

</code>

## Appendix A Things to do when changing versions

### Appendix A.1 Migrating from edmac to ledmac

If you have never used edmac, ignore this section. If you have used edmac and are starting on a completely new document, ignore this section. Only read this section if you are converting an original edmac document to use ledmac.

The package still provides the original `\text` command, but it is (a) deprecated, and (b) its name has been changed<sup>35</sup> to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext` Within numbered paragraphs, footnotes and endnotes are generated by forms of the `\critext` macro:

```
\critext{⟨lemma⟩}⟨commands⟩/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

<code>I saw my friend \critext{Smith}</code>	1 I saw my friend
<code>\Afootnote{Jones C, D.}/</code>	2 Smith on Tuesday.
on Tuesday.	<u>2 Smith]</u> Jones C, D.

The lemma `Smith` is printed as part of this sentence in the text, and is also made available to the footnote that specifies a variant, `Jones C, D`. The footnote macro is supplied with the line number at which the lemma appears in the main text.

The `⟨lemma⟩` may contain further `\critext` commands. Nesting makes it possible to print an explanatory note on a long passage together with notes on variants for individual words within the passage. For example:

<code>\critext{I saw my friend</code>	1 I saw my friend
<code>\critext{Smith}{\Afootnote{Jones</code>	2 Smith on Tuesday.
<code>C, D.}/ on Tuesday.}</code>	<u>2 Smith]</u> Jones C, D.
<code>\Bfootnote{The date was</code>	1-2 I saw my friend
<code>July 16, 1954.}</code>	Smith on Tuesday.] The
<code>/</code>	date was July 16, 1954.

However, `\critext` cannot handle overlapping but unnested notes—for example, one note covering lines 10–15, and another covering 12–18; a `\critext` that starts in the `⟨lemma⟩` argument of another `\critext` must end there, too. (The `\lemma` and `\linenum` commands may be used to generate overlapping notes if necessary.)

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

```
\catcode`\<=\active
```

<sup>35</sup>A name like `\text` is likely to be defined by other  $\text{\TeX}$  packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.



```
\let<=\critext
```

Then you might say `<{Smith}\variant{Jones}/`. This of course destroys the ability to use `<` in any new macro definitions, so long as it remains in effect; hence it should be used with care.

Changing the character at the end of the command requires more work:

```
\catcode`\<=\active
\def\xtext#1#2>{\critext{#1}{#2}/}
\let<=\xtext
```

This allows you to say `<{Smith}\Afootnote{Jones}>`.

Aliases for `\critext` of the first kind shown here also can't be nested—that is, you can't use the alias in the text that forms the first argument to `\critext`. (See VI p. 108 to find out why.) Aliases of the second kind may be nested without any problem.

If you really have to use `\critext` in any of the tabular or array environments, then `\edtext` must not be used in the same environment. If you use `\critext` in one of these environments then you have to issue the declaration `\usingcritext` beforehand. The declaration `\usingedtext` must be issued to revert to the default assumption that `\edtext` will be used.

## Appendix A.2 Migration from `ledmac` to `eledmac`

In `eledmac`, some changes were made in the code to allow easy customization. This may cause problems for people who have already made their own. The next sections explain how to handle this.

If you have created your own series using `\addfootins` and `\addfootinsX`, you must use instead the `\newseries` command (see 5.5.1 p. 29), and remove any `\Xfootnote` command.

If you have customized the `\XXXXXfmt` command, please check whether you can achieve the same by the commands documented for display options (6 p. 30) or `\Xfootnote` options (5.2.2 p. 23). Otherwise please add a new ticket on Github to request a new function for doing this.<sup>36</sup>

If for some reason you do not want to make the modifications to use the new functions of `eledmac`, you can continue using your own `\XXXXXfmt` command, but you must replace:

```
\renewcommand*{XXXXfmt}[3]
```

with

```
\renewcommandx*{XXXXfmt}[4][4=Z]
```

---

<sup>36</sup><https://github.com/maieul/ledmac/issues>

If you do not make that, you will get a spurious [X], where X is series letter.

If you used a `\protect` command inside a `\footnote` command inside a numbered section, you must change the `\protect` to `\noexpand`. Otherwise the command after the `\protect` will be discarded.

### Appendix A.3 Migration to eledmac 1.5.1

The version 1.5.1 corrects a bug in `stanzaindentsrepetition` (cf. 8.3 p. 42). This bug had two consequences:

1. `stanzaindentsrepetition` did not work when its value was greater than 2.
2. `stanzaindentsrepetition` worked wrong when its value was equal to 2.

So, if you used `stanzaindentsrepetition` with a value equal to 2, you had to change your `\setstanzaindents`. Explanation:

```
\setcounter{stanzaindentsrepetition}{2}
\setstanzaindents{5,1,0}
```

This code, in versions prior to 1.5.1, made the first line have an indentation of 0, the second line of 1, the third verse of 0, the fourth verse of 1 and so forth.

But this code should have instead achieved quite the contrary: the first line would have an indentation of 1, the second line of 0, the third line of 1, the fourth line of 0 and so forth.

So version 1.5.1 corrected this bug. If you want to keep the former presentation, you must change:

```
\setcounter{stanzaindentsrepetition}{2}
\setstanzaindents{5,1,0}
```

to:

```
\setcounter{stanzaindentsrepetition}{2}
\setstanzaindents{5,0,1}
```

### Appendix A.4 Migration to eledmac 1.12.0

The migration to eledmac 1.12.0 is easy:

- You must first delete all the auxiliary files, then compile your document three times as usual.
- If you have modified `\l@reg`, which is not advisable, you must rename it to `\@nl@reg`.

There is an additional problem. If you have put text into brackets just after `\pstart` or `\pend`, this text will be considered to be an optional argument of `\pstart` or `\pend` (see 4.2.3 p. 17). If so, add a `\relax` between `\pstart`/`\pend` and the first bracket.

The version 1.12.0 also introduces a better way to handle sectional divisions inside numbered text. Please read 15.2 p. 57.

## Appendix A.5 Migration to eledmac 17.1

This version changes the default setting of `\Xpstart`. Henceforth, `pstart` numbers will be printed in footnotes within the section of text where you have called `\numberpstarttrue`.

We do not see any reason to print them in the other sections. However, if you want to print the `pstart` numbers in all of the footnotes, whatever the section, without having to use `\numberpstarttrue`, you can use `\Xpstarteverytime`.

## Appendix A.6 Migration to eledmac 1.21.0

### Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX`

The `\ledsetnormalparstuff` has been split into two different commands:

- `\Xledsetnormalparstuff` for critical notes;
- `\ledsetnormalparstuffX` for familiar notes.

Both commands can take an optional argument which is the series letter. If you have redefined `\ledsetnormalparstuff` or any of the commands which call them, you must change them accordingly.

### Appendix A.6.2 Endnotes

In any case, delete the `.end` file before the next run.

The previous version of Eledmac had a bug: there were two spaces between the starting page number and the starting line number, but only one space between the ending page number and the ending line number.

As a matter of fact, a spurious space was added after the first `\printnpnum`. This spurious space has been deleted. However, if you want to keep the previous spurious space, you may load the package with the `oldprintnpnumspace` option.

If you have redefined `\endprint`, you must:

- Contact us and ask for the feature that required your hack, in order to avoid such a hack in the future.
- Use the new fifth argument.
- Add `\xdef\@currentseries{#4}` at the beginning of your own command.

## Appendix A.7 Migration to eledmac 1.22.0

The `\ledinnote` command now takes a first optional argument, which is the label for the hyperreference. If you have redefined it, change your redefinition, and check whether you can avoid this redefinition by only redefining `\ledinnotemark`.

## Appendix A.8 Migration to eledmac 1.23.0

You must delete the numbered auxiliary files before compiling with the new version of eledmac.

## Appendix A.9 Migration from eledmac to reledmac

There are many changes in reledmac which require the user to make modifications.

### Appendix A.9.1 Risk of ‘no room for a new’

The risk to obtain a ‘no room for a new something’ error is greater in reledmac than it is in eledmac. See 18.1.2 p. 60 in order to know how to limit it.

### Appendix A.9.2 Multiple indices with memoir

Eledmac and ledmac used the specific indexing tools of the memoir class designed to produce multiple indices. However, eledmac could also use imakeidx or indextools tools independently of the memoir class. This system forced to maintain redundant code. Since reledmac, we use only the imakeidx or indextools tools.

Consequently: Users of memoir are invited to use indextool or imakeidx to produce multiple indices.

### Appendix A.9.3 Deprecated commands and options

The table of deprecated commands and their alternatives follows. Note that the way some commands must be used may have changed. Please read the handbook.

<i>Deprecated command</i>	<i>Replaced with</i>
<code>\addfootins</code>	<code>\newseries</code>
<code>\addfootinsX</code>	<code>\newseries</code>
<code>\critext</code>	<code>\edtext</code>
<code>\falseverse</code>	<code>\newverse</code>
<code>\interparanoteglue</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\ledchapter</code>	<code>\eledchapter</code>
<code>\ledsection</code>	<code>\eledsection</code>
<code>\ledsetnormalparstuff</code>	<code>\Xledsetnormalparstuff</code> and <code>\ledsetnormalparstuffX</code>
<code>\ledsubsection</code>	<code>\eledsubsection</code>
<code>\ledsubsubsection</code>	<code>\eledsubsubsection</code>
<code>\noeledsec</code>	Package option <code>noeledsec</code>
<code>\noendnotes</code>	Package option <code>noendnotes</code>
<code>\pageparbreak</code>	<code>\ledpb</code>

The `ledsecnolinenum` option has been removed, because it was related to deprecated commands.

The `oldprintnppnumspace` option has been removed too, because it was related to a historical bug. The `\usingedtext` and `\usingcritext` commands are also deprecated.

**Appendix A.9.4 \renewcommand replaced by command**

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

<i>Deprecated \renewcommand</i>	<i>Replaced with</i>
\@led@extranofeet	\newseries
\apprefprefixmore	\setapprefprefixmore
\apprefprefixsingle	\setapprefprefixsingle
\endstanzaextra	Optional argument of \&
\hangingsymbol	\sethangingsymbol
\ledfootinsdim	\Xmaxhnotes and \maxhnotesX
\parafootftmsep	\Xparafootsep and \parafootsepX
\notenumfont	\Xnotenumfont, \Xendnotenumfont and \notenumfontX
\notefontsetup	\Xnotefontsize, \Xendnotefontsize and \notefontsizeX
\sidenoteseq	\setsidenotsep
\startstanzahook	Optional argument of \stanza
\symplinenum	\Xsymplinenum

**Appendix A.9.5 Commands the names of which have been changed**

In order to help the migration from eledmac to reledmac, you may load reledmac with eledmac-compat option. However, it is advised not to, and to change the command names themselves instead. In many cases, you use only a few of them, except the \footparagraph command.

<i>Old command</i>	<i>New command</i>
\footparagraph	\Xarrangement
\footnormal	\Xarrangement
\foottwocol	\Xarrangement
\footthreecol	\Xarrangement
\footparagraphX	\arrangementX
\footnormalX	\arrangementX
\foottwocolX	\arrangementX
\footthreecolX	\arrangementX
\afterlemmaseparator	\Xafterlemmaseparator
\afternote	\Xafternote and \afternoteX
\afternumberinfootnote	\Xafternumber
\afterXrule	\Xafterrule
\afterXsymplinenum	\Xaftersymplinenum
\beforelemmaseparator	\Xbeforelemmaseparator
\beforenumberinfootnote	\Xbeforenumber
\beforeXnotes	\Xbeforenotes
\beforeXsymplinenum	\Xbeforesymplinenum

<i>Old command</i>	<i>New command</i>
<code>\bhookXnote</code>	<code>\Xbhookendnote</code>
<code>\bhookXnote</code>	<code>\Xbhooknote</code>
<code>\boxendlinenum</code>	<code>\Xboxendlinenum</code>
<code>\boxlinenum</code>	<code>\Xboxlinenum</code>
<code>\boxlinenumalign</code>	<code>\Xboxlinenumalign</code>
<code>\boxstartlinenum</code>	<code>\Xboxstartlinenum</code>
<code>\boxsymlinenum</code>	<code>\Xboxsymlinenum</code>
<code>\boxXendlinenum</code>	<code>\Xendboxlinenum</code>
<code>\boxXendlinenumalign</code>	<code>\Xendboxlinenumalign</code>
<code>\boxXendstartlinenum</code>	<code>\boxXendstartlinenum</code>
<code>\letboxXendendlinenum</code>	<code>\Xendletboxendlinenum</code>
<code>\hsizetwocol</code>	<code>\Xhsizetwocol</code>
<code>\hsizethreecol</code>	<code>\Xhsizethreecol</code>
<code>\inplaceoflemmaseparator</code>	<code>\Xinplaceoflemmaseparator</code>
<code>\inplaceofnumber</code>	<code>\Xinplaceofnumber</code>
<code>\lemmaseparator</code>	<code>\Xlemmaseparator</code>
<code>\maxhXnotes</code>	<code>\Xmaxhnotes</code>
<code>\morethantwolines</code>	<code>\Xmorethantwolines</code>
<code>\nonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\notesXwidthliketwocolumns</code>	<code>\noteswidthliketwocolumnsX</code>
<code>\noXlemmaseparator</code>	<code>\Xnolemmaseparator</code>
<code>\numberonlyfirstinline</code>	<code>\Xnumberonlyfirstinline</code>
<code>\numberonlyfirstintwolines</code>	<code>\Xnumberonlyfirstintwolines</code>
<code>\nonbreakableafternumber</code>	<code>\Xnonbreakableafternumber</code>
<code>\onlyXpstart</code>	<code>\Xonlypstart</code>
<code>\parafootsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\pstartinfootnote</code>	<code>\Xpstart</code>
<code>\pstartinfootnoteeverytime</code>	<code>\Xpstarteverytime</code>
<code>\symlinenum</code>	<code>\Xsymlinenum</code>
<code>\twolines</code>	<code>\Xtwolines</code>
<code>\twolinesbutnotmore</code>	<code>\Xtwolinesbutnotmore</code>
<code>\twolinesonlyinsamepage</code>	<code>\Xtwolinesonlyinsamepage</code>
<code>\txtbeforeXnotes</code>	<code>\Xtxtbeforenotes</code>
<code>\XendXafterlemmaseparator</code>	<code>\Xendafterlemmaseparator</code>
<code>\XendXbeforelemmaseparator</code>	<code>\Xendbeforelemmaseparator</code>
<code>\XendXinplaceoflemmaseparator</code>	<code>\Xendinplaceoflemmaseparator</code>
<code>\XendXlemmaseparator</code>	<code>\Xendlemmaseparator</code>
<code>\XendXmorethantwolines</code>	<code>\Xendmorethantwolines</code>
<code>\XendXtwolines</code>	<code>\Xendtwolines</code>
<code>\Xnonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\lineref</code>	<code>\edlineref</code>

### Appendix A.9.6 Endnotes

With reledmac, there is now one auxiliary file for every endnotes set (.Aend, .Bend, .Cend etc.). If you have overridden \doendnotes (which you would not have done) you must adapt your code.

### Appendix A.9.7 Z Series

The ‘Z’ series of notes has been removed. Only five series are provided now by default: A, B, C, D, E.

### Appendix A.9.8 Internal commands

Users who have overridden internal commands, which is wrong, must adapt according to the following. Or better, they should not override any of such commands and use reledmac options instead.

- If you have modified \Xfootfmt, note that the fourth argument is now mandatory.
- \unvxh has been replaced with \Xunvxh and \unvxhX with two mandatory arguments.

## Appendix A.10 Migration to reledmac 2.1.0

Reledmac 2.1.0 fix some bugs when using \Xbhooknote and \bhooknoteX not in order to execute code at the beginning of each notes, but to insert content of at the beginning of each notes.

People who use these commands to do it, which is not the original idea, must change the following:

1. Horizontal space is no longer automatically added after the content of the \Xbhooknote/\bhooknoteX argument. You must include it manually. So instead of \Xbhooknote{content}, use \Xbhooknote{content }.
2. Indent is no longer automatically added before the content of the \Xbhooknote/\bhooknoteX argument. If you want to keep it, add \indent in the argument of \Xbhooknote/\bhooknoteX.

## Appendix A.11 Migration to reledmac 2.1.3

Reledmac 2.1.3 fix an historical bug, (style in ledmac 0.7!) which doubled the space before the rules of paragraphed familiar footnotes. Consequently, if you use paragraphed familiar footnotes, you should maybe adapt it, playing with \beforenotesX.

## Appendix A.12 Migration to reledmac 2.3.0

Before reledmac 2.3.0, for typesetting verse, any empty line was considered a paragraph inside verses. Counting empty lines this created breaking verse, hanging verses, and also added spurious vertical spaces. Version 2.3.0 disables paragraph in stanza. If you want vertical space, use optional argument of \stanza or \endverse.

### Appendix A.13 Migration to reledmac 2.4.0

It is not mandatory, but strongly recommended, to change any `\renewcommand{\endashchar}{\langle...\rangle}` to the use of `\Xlinerangeseparator` or `/` and `\Xendlinerangeseparator` (6.2.3 p. 31).

### Appendix A.14 Migration to reledmac 2.5.0

It is strongly recommended to stop redefining `\printnnum` and to use the hooks documented in 6.3 p. 35.

`\xlineref` does not print anymore the side flag (R for right side), because it is incompatible with numerical test. Use `\xflagref` to obtain it.

The `\printlines` and `\printendlines` commands take now an eighth argument, which is the side flag. It is strongly recommended to NEVER redefine these two commands and to use the setting commands instead (or to ask for new setting commands if the actual does not answer to your needs). However, if you have done it, just change your redefinition to have a new argument.

It is strongly recommended to stop redefining `\fullstop` and to use `\Xsublinesep` instead.

### Appendix A.15 Migration to reledmac 2.7.0

`\SErefonlypage` (introduced in reledmac 2.5.0) added an parenthesis after the page number. This was just an error, linked to a bad imitation of `\SErefwithpage`. That has been deleted. And so, the `\XendafterpagenumberSErefonlypage` to set it was also deleted.

`\rigidbalance` is split to two new commands: `\Xrigidbalance` for critical footnotes and `\rigidbalanceX` for familiar footnotes. If you have redefined it – but why should you have ?–, you should split your single redefinition in two redefinitions.

### Appendix A.16 Migration to reledmac 2.7.2

`\Xhsize` is already defined in the floatrow package. It becomes `\Xwidth`, and, consequently, `\hsizeX` becomes `\widthX`.

The ancient names are temporarily maintained as aliases.

### Appendix A.17 Migration to reledmac 2.8.0

Reledmac 2.8.0 fix spurious indents for paragraphed critical and familiar footnotes in `ledgroup` and `minipage`. You can re-establish the indent with `\Xparinden` and `\parindentX`.



## References

- [Bre96] Herbert Breger. `tabmac`. October 1996. (Available from CTAN in `macros/plain/contrib/tabmac`)
- [Bur01] John Burt. ‘Typesetting critical editions of poetry’. *TUGboat*, **22**, 4, pp 353–361, December 2001. (Code available from CTAN in `macros/latex/contrib/poemscol`)
- [Eck03] Matthias Eckermann. *The Parallel-Package*. April 2003. (Available from CTAN in `macros/latex/contrib/parallel`)
- [Fai03] Robin Fairbairns. *footmisc—a portmanteau package for customising footnotes in L<sup>A</sup>T<sub>E</sub>X*. February 2003. (Available from CTAN in `macros/latex/contrib/footmisc`)
- [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`)
- [Lüc03] Uwe Lück. ‘`ednotes`—critical edition typesetting with LaTeX’. *TUGboat*, **24**, 2, pp. 224–236, 2003. (Code available from CTAN in `macros/latex/contrib/ednotes`)
- [Sul92] Wayne G. Sullivan. *The file `edstanz.doc`*. June 1992. (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. *Parallel typesetting for critical editions: the `eledpar` package*. December 2004. (Available from CTAN in `macros/latex/contrib/ledmmac`)

## Index

### Symbols

<code>\&amp;</code> .....	41
<code>\@EDROWFILL@</code> .....	1
<code>\@adv</code> .....	1
<code>\@advancestanzanumber</code> .....	1
<code>\@doclearpage</code> .....	1
<code>\@doreinfeetX</code> .....	1
<code>\@edindex@hyperref</code> .....	1
<code>\@edrowfill@</code> .....	1
<code>\@edtext@level</code> .....	1
<code>\@emptytoks</code> .....	1
<code>\@fnpos</code> .....	1

<code>\@footnotemark</code> .....	1
<code>\@footnotetext</code> .....	1
<code>\@getfirstseries</code> .....	1
<code>\@gobblefive</code> .....	1
<code>\@gobblefour</code> .....	1
<code>\@gobblethree</code> .....	1
<code>\@h</code> .....	1
<code>\@changingsymbol</code> .....	1
<code>\@iiminipage</code> .....	1
<code>\@insertstanzanumber</code> .....	1
<code>\@k</code> .....	1
<code>\@l@dttempcnta</code> .....	1
<code>\@l@dttempcntb</code> .....	1
<code>\@lab</code> .....	1
<code>\@led@testifnofoot</code> .....	1
<code>\@lemma</code> .....	1
<code>\@line@#num</code> .....	1
<code>\@lock</code> .....	1
<code>\@lopL</code> .....	1
<code>\@lopR</code> .....	1
<code>\@makechapterhead</code> .....	1
<code>\@makeschapterhead</code> .....	1
<code>\@mem@extranofeet</code> .....	1
<code>\@mem@old@ssect</code> .....	1
<code>\@mpfnpos</code> .....	1
<code>\@nl</code> .....	1
<code>\@nl@reg</code> .....	1
<code>\@opXfeet</code> .....	1
<code>\@pend</code> .....	1
<code>\@pendR</code> .....	1
<code>\@ref</code> .....	1
<code>\@ref@reg</code> .....	1
<code>\@sect</code> .....	1
<code>\@series</code> .....	1
<code>\@set</code> .....	1
<code>\@sidenoteseq</code> .....	1
<code>\@ssect</code> .....	1
<code>\@startstanza</code> .....	1
<code>\@stopstanza</code> .....	1
<code>\@sw</code> .....	1
<code>\@tag</code> .....	1
<code>\@wredindex</code> .....	1
<code>\@xloop</code> .....	1
<code>\@xympar</code> .....	1
CLASSbook .....	297
CLASSmemoir .....	175, 229, 230, 259, 297, 298, 316, 370, 374
CLASSscrbook .....	374
COMMAND\*footnote .....	61
COMMAND\...@footnotemark... .....	178
COMMAND\...d@ta .....	131

COMMAND\ <code>&lt;hook</code>	
<code>@&lt;series</code> . . . . .	218
COMMAND\ <code>&lt;hookname</code>	
<code>&lt;pseudoserries</code> . . . . .	220
COMMAND\ <code>&lt;type</code>	
<code>footfmt</code> . . . . .	166
COMMAND\ <code>@@line</code> . . . . .	159
COMMAND\ <code>@MM</code> . . . . .	146, 371
COMMAND\ <code>@Rlineflag</code> . . . . .	261, 371
COMMAND\ <code>@SErefprefix</code> . . . . .	241
COMMAND\ <code>@SErefprefixmore</code> . . . . .	241
COMMAND\ <code>@add@</code> . . . . .	287
COMMAND\ <code>@adv</code> . . . . .	96
COMMAND\ <code>@apprefprefixmore</code> . . . . .	241
COMMAND\ <code>@apprefprefixsingle</code> . . . . .	241
COMMAND\ <code>@bsphack</code> . . . . .	232
COMMAND\ <code>@doclearpage</code> . . . . .	230, 364, 374
COMMAND\ <code>@doreinfeetX</code> . . . . .	374
COMMAND\ <code>@dprintingcolumns</code> . . . . .	371
COMMAND\ <code>@edindex@hyperref</code> . . . . .	261, 262
COMMAND\ <code>@edtext@level</code> . . . . .	112
COMMAND\ <code>@esphack</code> . . . . .	232
COMMAND\ <code>@fnpos</code> . . . . .	196
COMMAND\ <code>@footnotemark</code> . . . . .	175, 176, 364, 374
COMMAND\ <code>@footnotetext</code> . . . . .	175, 176, 364
COMMAND\ <code>@gobble</code> . . . . .	110
COMMAND\ <code>@gobblefive</code> . . . . .	215, 372
COMMAND\ <code>@gobblefour</code> . . . . .	370
COMMAND\ <code>@gobblethree</code> . . . . .	363
COMMAND\ <code>@h</code> . . . . .	161
COMMAND\ <code>@hangingsymbol</code> . . . . .	265
COMMAND\ <code>@iiiminipage</code> . . . . .	251, 253, 363, 374
COMMAND\ <code>@iiiminpage</code> . . . . .	251
COMMAND\ <code>@l</code> . . . . .	369
COMMAND\ <code>@l@tempcnta</code> . . . . .	134, 135, 142
COMMAND\ <code>@l@tempcntb</code> . . . . .	135
COMMAND\ <code>@l@reg</code> . . . . .	369
COMMAND\ <code>@lab</code> . . . . .	92, 231, 234, 237, 363
COMMAND\ <code>@ldunboxmpfoot</code> . . . . .	253
COMMAND\ <code>@led@extranofeet</code> . . . . .	317
COMMAND\ <code>@ledinnote@command</code> . . . . .	257, 258
COMMAND\ <code>@lemma</code> . . . . .	115, 116
COMMAND\ <code>@lock</code> . . . . .	87, 266
COMMAND\ <code>@lopL</code> . . . . .	364
COMMAND\ <code>@lopR</code> . . . . .	364
COMMAND\ <code>@makecol</code> . . . . .	226, 227, 229, 374
COMMAND\ <code>@mpfnpos</code> . . . . .	196
COMMAND\ <code>@nl</code> . . . . .	92, 93, 95, 97, 105, 234, 363, 364
COMMAND\ <code>@nl@reg</code> . . . . .	93, 314, 364, 369
COMMAND\ <code>@opXfeet</code> . . . . .	364

COMMAND\@opfeetX	374
COMMAND\@opxtrafeeti	374
COMMAND\@page	95, 234
COMMAND\@pend	364
COMMAND\@pendR	364
COMMAND\@ref	92, 101–103, 106, 110
COMMAND\@ref@reg	101, 364
COMMAND\@reinserts	226–229, 374
COMMAND\@secondoftwo	62
COMMAND\@sect	300
COMMAND\@series	216
COMMAND\@set	96
COMMAND\@sidenoteseq	250
COMMAND\@sw	102, 103, 118, 121
COMMAND\@tag	111, 113, 116
COMMAND\@tempcnta	73
COMMAND\@tempcntb	73
COMMAND\@toksa	79
COMMAND\@toksb	79
COMMAND\@xloop	143
COMMAND\@xympar	245, 374
COMMAND\Aendnote	15, 23
COMMAND\Afootfmt	146
COMMAND\Afootgroup	146
COMMAND\Afootnote	8, 15, 22, 23, 25, 113, 153, 175, 197, 209, 373
COMMAND\Afootstart	146
COMMAND\AtEveryPend	17, 127, 370, 372, 374
COMMAND\AtEveryPstart	17, 370, 372, 374, 377
COMMAND\Bendnote	15, 22
COMMAND\Bfootnote	8, 15, 175, 197, 209
COMMAND\Centering	38
COMMAND\Cfootnote	175
COMMAND\Columns	73, 150
COMMAND\Dfootnote	175
COMMAND\Efootnote	175
COMMAND\Gls	53
COMMAND\Hy@raisedlink	309
COMMAND\Hy@raisedlink@left	309
COMMAND\NR	300
COMMAND\Pages	73, 227, 228
COMMAND\ProcessOptionsX	65
COMMAND\RaggedLeft	38
COMMAND\RaggedRight	38
COMMAND\SEonlypage	239, 376
COMMAND\SEref	47–49, 239, 241, 377
COMMAND\SErefonlypage	48, 49, 320, 376
COMMAND\SErefwithpage	47–49, 239, 241, 320, 376
COMMAND\Stanza	369
COMMAND\Waklam	288
COMMAND\X@doreinfeet	228, 374

COMMAND\XR@prefix	244
COMMAND\XR@test	244
COMMAND\XR@test@mac	244
COMMAND\XR@test@mac@test	244
COMMAND\XXXXXXfmt	313
COMMAND\XXXXXfmt	313
COMMAND\Xafterlemmaseparator	35, 317
COMMAND\Xafternote	38, 316, 317
COMMAND\Xafternumber	33, 317
COMMAND\Xafterrule	39, 197, 317, 369, 372
COMMAND\Xaftersymmlinum	33, 317
COMMAND\Xarrangement	30, 61, 147, 218, 317
COMMAND\Xarrangement@footparagraph	152
COMMAND\Xarrangement@normal	147
COMMAND\Xarrangement@paragraph	151
COMMAND\Xbeforelemmaseparator	35, 317
COMMAND\Xbeforenotes	39, 196, 197, 317, 369, 372
COMMAND\Xbeforenumber	31, 33, 317
COMMAND\Xbeforesymmlinum	33, 317
COMMAND\Xbhookendnote	318
COMMAND\Xbhookgroup	39, 376, 377
COMMAND\Xbhooknote	37, 318, 319, 374, 375
COMMAND\Xboxendlinum	34, 318, 373
COMMAND\Xboxlininum	34, 318
COMMAND\Xboxlininumalign	34, 318, 373
COMMAND\Xboxstartlininum	34, 318, 373
COMMAND\Xboxsymmlinum	34, 318
COMMAND\Xcolalign	37, 372
COMMAND\Xdo@feet	227, 364, 374
COMMAND\Xend	215
COMMAND\XendXafterlemmaseparator	318
COMMAND\XendXbeforelemmaseparator	318
COMMAND\XendXinplaceoflemmaseparator	318
COMMAND\XendXlemmaseparator	318
COMMAND\XendXmorethantwolines	318
COMMAND\XendXtwolines	318
COMMAND\Xendafterenumber	33, 375
COMMAND\Xendafterlemmaseparator	36, 318
COMMAND\Xendafternote	40, 377
COMMAND\Xendafternumber	35
COMMAND\Xendafterpagenumber	35, 49
COMMAND\XendafterpagenumberSErefonlypage	320
COMMAND\Xendaftersymmlinum	33, 35, 375
COMMAND\Xendahookinplaceofnumber	35, 375
COMMAND\Xendahooklininum	35, 375
COMMAND\Xendbeforelemmaseparator	36, 318
COMMAND\Xendbeforelininum	35
COMMAND\Xendbeforenumber	33, 375
COMMAND\Xendbeforepagenumber	35, 48, 49
COMMAND\XendbeforepagenumberSErefonlypage	48

COMMAND\Xendbeforesymlinewidth	33, 35, 375
COMMAND\Xendbhookinplaceofnumber	35, 375
COMMAND\Xendbhooklinewidth	35, 375
COMMAND\Xendbhooknote	37
COMMAND\Xendboxendlinewidth	34, 373
COMMAND\Xendboxlinewidth	34, 318, 371
COMMAND\Xendboxlinewidthalign	34, 318, 373
COMMAND\Xendboxstartlinewidth	34, 373
COMMAND\Xendboxsymlinewidth	34, 375
COMMAND\Xendhangindent	37, 375, 377
COMMAND\Xendinplaceoflemmaseparator	23, 36, 318
COMMAND\Xendinplaceofnumber	34, 374
COMMAND\Xendinsertsep@	202
COMMAND\Xendlemmadisablefontselection	36
COMMAND\Xendlemmafnt	36, 376
COMMAND\Xendlemmaseparator	24, 36, 318
COMMAND\Xendletboxendlinewidth	318
COMMAND\Xendlineflag	49
COMMAND\Xendlineprefixmore	35, 49
COMMAND\Xendlineprefixsingle	35, 49
COMMAND\Xendlinerrangeseparator	31, 49, 320, 375
COMMAND\Xendmorethantwolines	23, 32, 49, 318, 372, 373
COMMAND\Xendnonumber	32, 374
COMMAND\Xendnote	200, 214, 215, 372
COMMAND\Xendnotefontsize	37, 317
COMMAND\Xendnotenumfont	35, 36, 317
COMMAND\Xendnumberonlyfirstinline	31, 375
COMMAND\Xendnumberonlyfirstintwolines	31, 375
COMMAND\Xendparagraph	40, 369
COMMAND\Xendsep	40
COMMAND\Xendsublinesep	33, 49
COMMAND\Xendsymlinewidth	31, 375
COMMAND\Xendtwolines	23, 32, 49, 318, 372, 373
COMMAND\Xendtwolinesbutnotmore	32, 49, 372, 373
COMMAND\Xendtwolinesonlyinsamepage	32, 49, 372, 373
COMMAND\Xfootfmt	319
COMMAND\Xfootgroup	151
COMMAND\Xfootins	149
COMMAND\Xfootnote	46, 52, 111, 313, 366, 370–372, 376
COMMAND\Xfootstarts	151
COMMAND\Xhangindent	37, 375
COMMAND\Xhsize	320, 376, 377
COMMAND\Xhsizethreecol	38, 40, 318
COMMAND\Xhsizetwocol	38, 40, 220, 318
COMMAND\Xinplaceoflemmaseparator	23, 35, 318
COMMAND\Xinplaceofnumber	34, 318, 371, 373
COMMAND\Xinsertparafootsep	156, 158
COMMAND\Xledsetnormalparstuff	315, 316, 372
COMMAND\Xlemmadisablefontselection	36
COMMAND\Xlemmafnt	36, 376

COMMAND\Xlemmaseparator	35, 221, 223, 226, 318
COMMAND\Xlineflag	49
COMMAND\Xlinerangeseparator	31, 49, 320, 375
COMMAND\Xmaxhnotes	39, 61, 317, 318, 369, 371
COMMAND\Xmorethantwolines	23, 32, 49, 318, 371
COMMAND\Xnoindent	375
COMMAND\Xnolemmaseparator	35, 226, 318
COMMAND\Xnonbreakableafternumber	33, 318, 367
COMMAND\Xnonumber	32, 318
COMMAND\Xnonumberinfootnote	318
COMMAND\Xnotefontsize	37, 317
COMMAND\Xnotefontsize@<s>	157, 160, 161
COMMAND\Xnotenumfont	36, 317
COMMAND\Xnoteswidthliketwocolumns	40, 370
COMMAND\Xnumberonlyfirstinline	31, 90, 220, 221, 223, 318, 366, 371
COMMAND\Xnumberonlyfirstintwolines	31, 318, 366
COMMAND\Xonlypstart	33, 318, 366, 371
COMMAND\Xparafootsep	38, 90, 317, 318
COMMAND\Xparafootsep@series	156
COMMAND\Xparinden	320
COMMAND\Xparindent	37, 372, 375, 377
COMMAND\Xpstart	32, 33, 315, 318, 366, 371
COMMAND\Xpstarteverytime	32, 315, 318, 371
COMMAND\Xragged	38
COMMAND\Xrigidbalance	159, 320, 376
COMMAND\Xstanza	33, 44
COMMAND\Xstanzaseparator	33
COMMAND\Xsublinesep	20, 33, 49, 320
COMMAND\Xsublinesepside	20, 33
COMMAND\Xsymlinenum	31, 38, 317, 318, 373
COMMAND\Xtwolines	23, 32, 49, 171, 172, 220, 318, 371
COMMAND\Xtwolinesappref	220
COMMAND\Xtwolinesbutnotmore	32, 49, 318, 372
COMMAND\Xtwolinesbutnotmoreappref	220
COMMAND\Xtwolinesonlyinsamepage	32, 49, 318, 372
COMMAND\Xtxtbeforenotes	39, 318
COMMAND\Xunvxh	154, 319
COMMAND\Xwidth	40, 320, 377
COMMAND\&	317
COMMAND\absline@num	86, 87, 133
COMMAND\accent	110
COMMAND\actionlines@list	88, 134
COMMAND\actions@list	88
COMMAND\add@inserts	88, 141
COMMAND\add@inserts@next	141
COMMAND\add@penalties	133, 142
COMMAND\addcontentsline	295
COMMAND\addfootins	313, 316
COMMAND\addfootinsX	313, 316
COMMAND\advancelabel@refs	233

COMMAND\advanceline	21, 89, 96, 107, 374
COMMAND\advancepageno	226
COMMAND\affixlin@num	250
COMMAND\affixline@num	135, 138, 139, 364
COMMAND\affixpstart@num	139
COMMAND\afterXrule	317
COMMAND\afterXsymlinenum	317
COMMAND\afterrenumber	33
COMMAND\aftergroup	110, 114
COMMAND\afterlemmaseparator	317
COMMAND\afternote	317
COMMAND\afternoteX	38, 316, 317
COMMAND\afternumberinfootnote	317
COMMAND\afterruleX	39, 369, 372
COMMAND\applabel	48, 234–236, 372
COMMAND\appref	46, 48, 49, 239, 241, 376, 377
COMMAND\apprefprefixmore	48, 317
COMMAND\apprefprefixsingle	48, 317
COMMAND\apprefwithpage	48, 49, 239, 241, 373, 376
COMMAND\arrangementX	30, 61, 179, 218, 317
COMMAND\arrangementX@normal	184
COMMAND\at@every@pend	127
COMMAND\autopar	16, 17, 124, 127, 128, 194, 365, 367, 368, 372
COMMAND\ballast	60
COMMAND\ballast@count	133, 142
COMMAND\baselineskip	30, 152, 157
COMMAND\beforeXnotes	317
COMMAND\beforeXsymlinenum	317
COMMAND\beforeelectedchapter	9, 58, 296
COMMAND\beforelemmaseparator	317
COMMAND\beforenotesX	39, 319, 368, 369, 372
COMMAND\beforenumberinfootnote	317
COMMAND\begin	273
COMMAND\beginnumbering	15–18, 74, 75, 77, 86, 90, 91, 104, 127, 199, 366, 369, 373, 374
COMMAND\bf	366
COMMAND\bfseries	36, 366
COMMAND\bhookXnote	318
COMMAND\bhookgroupX	39, 376
COMMAND\bhooknoteX	37, 319, 374, 375
COMMAND\body	266
COMMAND\bodyfootmarkA	28
COMMAND\boxXendlinenum	318
COMMAND\boxXendlinenumalign	318
COMMAND\boxXendstartlinenum	318
COMMAND\boxendlinenum	318
COMMAND\boxlinefootnote	168
COMMAND\boxlinenum	318
COMMAND\boxlinenumalign	318
COMMAND\boxstartlinenum	318
COMMAND\boxsymlinenum	318



COMMAND\break	31, 154
COMMAND\brokenpenalty	142
COMMAND\centering	38
COMMAND\ch@ck@l@ck	365
COMMAND\ch@cksub@l@ck	138, 365
COMMAND\chapter	57, 298, 369, 372, 374
COMMAND\chaptermark	295
COMMAND\check@pb@in@verse	308
COMMAND\colalignX	37, 372
COMMAND\collect@body	273
COMMAND\colorbox	61
COMMAND\columns	40
COMMAND\columnwidth	152, 370
COMMAND\command names	220
COMMAND\copyright	111
COMMAND\correct@Xfootins@box	371
COMMAND\correct@footinsX@box	371
COMMAND\count	160
COMMAND\critex	365
COMMAND\critext	117, 312, 313, 316
COMMAND\curname	65, 120
COMMAND\ctab	289, 294
COMMAND\ctabtext	294
COMMAND\dcoll	282
COMMAND\def	63
COMMAND\detokenize	120
COMMAND\dimen	160
COMMAND\dimexpr	40
COMMAND\discretionary	154
COMMAND\displaywidowpenalty	142
COMMAND\do@actions	133, 134, 365
COMMAND\do@actions@fixedcode	364
COMMAND\do@actions@next	133, 134
COMMAND\do@ballast	133, 142
COMMAND\do@feetX	374
COMMAND\do@insidelinehook	367
COMMAND\do@line	88, 109, 125, 129, 131, 132, 141, 142, 266, 365, 367, 369
COMMAND\do@linehook	365
COMMAND\do@lockoff	89
COMMAND\do@lockon	89
COMMAND\dodoreintrafeet	363
COMMAND\doendnotes	24, 202, 203, 319, 372
COMMAND\doendnotesbysection	24, 203, 215, 373
COMMAND\doinsidelinehook	21, 370
COMMAND\dolinehook	21, 370
COMMAND\doreintrafeeti	374
COMMAND\doreintrafeetii	374
COMMAND\doxtrafeet	226, 363
COMMAND\doxtrafeeti	374
COMMAND\doxtrafeetii	374

COMMAND\dummy@ref	110
COMMAND\edaftertab	56, 288–290
COMMAND\edatleft	56, 286
COMMAND\edatright	56, 287
COMMAND\edbeforetab	56, 288, 289
COMMAND\edfilldimen	287
COMMAND\edfont@info	116
COMMAND\edgls	53, 256
COMMAND\edgls...	377
COMMAND\edindex	51–53, 256, 257, 260, 262, 277, 367, 370, 371, 374, 375
COMMAND\edindexlab	53
COMMAND\edlabel	45–48, 50, 111, 231, 233, 234, 236, 237, 244, 256, 277, 309, 363, 366–368, 371, 376
COMMAND\edlabelE	47, 236
COMMAND\edlabelS	47, 236
COMMAND\edlabelSE	47
COMMAND\edlineref	46, 231, 318, 371, 373, 376
COMMAND\edmakelabel	47, 244
COMMAND\edpageref	46, 231, 236, 244
COMMAND\edrowfill	288
COMMAND\edtabcolsep	282
COMMAND\edtext	6, 22–25, 27, 28, 41, 46–48, 51, 54, 61, 87, 101, 103, 105, 106, 108–119, 121, 122, 234, 235, 238, 277, 279, 298, 309, 312, 313, 316, 364, 365, 367, 369–373
COMMAND\edtext@level	373
COMMAND\edvertdots	57, 287
COMMAND\edvertline	56, 57, 287
COMMAND\elechapter	58
COMMAND\eled@sectioning@out	302
COMMAND\eledchapter	57, 316, 370, 374
COMMAND\eledchapter*	57
COMMAND\eledmac@error	363
COMMAND\eledsection	6, 15, 57, 110, 131, 296, 316, 371
COMMAND\eledsection*	57
COMMAND\eledsubsection	57, 316
COMMAND\eledsubsection*	57
COMMAND\eledsubsubsection	57, 316
COMMAND\eledsubsubsection*	57
COMMAND\eledxxx	9, 58, 297, 302, 369
COMMAND\eledxxxx	295
COMMAND\else	255, 296
COMMAND\empty	73, 136, 137, 231
COMMAND\end	272, 273
COMMAND\end@lemmas	110
COMMAND\endashchar	41, 165
COMMAND\endgraf	125, 156, 194
COMMAND\endlock	20, 89, 108, 270
COMMAND\endminipage	251, 253, 363, 374
COMMAND\endnotes	372, 376, 377
COMMAND\endnumbering	15, 18, 74, 76, 77, 365, 373
COMMAND\endprint	200, 202, 215, 315
COMMAND\endstanzaextra	317

COMMAND\endsub	20, 88, 106
COMMAND\endverse	319
COMMAND\everypar	128
COMMAND\extensionchars	59, 74
COMMAND\externaldocument	50, 244
COMMAND\fix@l@cks	365
COMMAND>falseverse	316, 367, 369
COMMAND\fi	296
COMMAND\firstlinenum	19, 135, 365
COMMAND\firstsublinenum	19, 365
COMMAND\fix@page	93, 95, 364
COMMAND\flag@end	106, 115, 369
COMMAND\flag@start	106, 115, 369, 370
COMMAND\flagstanza	44
COMMAND\floatingpenalty	146, 371
COMMAND\flush@notes	142, 143
COMMAND\fnpos	196, 368
COMMAND\footfmt	145, 148
COMMAND\footfmt...	179
COMMAND\footfootmarkA	28
COMMAND\footfudgefactor	154
COMMAND\footfudgefiddle	61, 152, 363
COMMAND\footgroup	146
COMMAND\footins	150
COMMAND\footnormal	219, 317, 364
COMMAND\footnormalX	317
COMMAND\footnote	28, 60, 175, 176, 314, 364
COMMAND\footnote@lang	165
COMMAND\footnoteA	15, 28
COMMAND\footnoteB	15
COMMAND\footnoteC	22
COMMAND\footnoteE	28
COMMAND\footnoteX	8, 212
COMMAND\footnoteX@reading	213
COMMAND\footnoteXmk	225
COMMAND\footnotelang@lua	144
COMMAND\footnotelang@poly	145
COMMAND\footnoteoption@	144, 375
COMMAND\footnoterule	160
COMMAND\footnotesize	37
COMMAND\footparagraph	152, 219, 317, 369
COMMAND\footparagraphX	189, 317, 369
COMMAND\footsplitskips	365, 371
COMMAND\footstart	146, 149, 160
COMMAND\footstrut	156
COMMAND\footthreecol	317
COMMAND\footthreecolX	317, 372
COMMAND\foottwocol	317
COMMAND\foottwocolX	317, 372
COMMAND\fullstop	41, 320

COMMAND\get@edindex@hyperref	261
COMMAND\get@edindex@ledinnote@command	257
COMMAND\get@fnmark	176
COMMAND\get@index@command	368
COMMAND\get@linelistfile	365
COMMAND\get@thisfootnote	183
COMMAND\getline@num	132, 134
COMMAND\gl@p	79
COMMAND\global	92
COMMAND\globaldefs	92
COMMAND\gls	53, 264
COMMAND\hangindentX	37, 372, 375
COMMAND\hangingsymbol	317, 365
COMMAND\hbox	154
COMMAND\hfill	368
COMMAND\hidenumbering	21, 101, 372
COMMAND\hline	54
COMMAND\hrulefill	288
COMMAND\hsize	31, 150, 152–154, 160, 163, 194, 364, 370
COMMAND\hsizeX	320, 376, 377
COMMAND\hsizethreecol	318
COMMAND\hsizethreecolX	38, 40
COMMAND\hsizetwocol	318
COMMAND\hsizetwocolX	38, 40
COMMAND\hyperlinkR	260
COMMAND\hyperlinkformat	260
COMMAND\hyperlinkformatR	261
COMMAND\if@RTL	67
COMMAND\if@edtext@	370, 373
COMMAND\if@eled@sectioning	297
COMMAND\if@nobreak	126
COMMAND\if@noneed@Footnote	105
COMMAND\ifXnote@	74
COMMAND\ifbypage@	79
COMMAND\ifbypage@R	80
COMMAND\ifbypstart@	79
COMMAND\ifbypstart@R	80
COMMAND\iffirst@linenum@out@	104
COMMAND\ifindtl@innote	74
COMMAND\ifindtl@notenumber	74
COMMAND\ifinserthangingsymbol	266
COMMAND\ifinstanza	266
COMMAND\ifstwofollowinglines	172
COMMAND\ifl@d@Xmorethantwolines	169, 372
COMMAND\ifl@d@Xtwolines	169
COMMAND\ifl@d@dash	169
COMMAND\ifl@d@elin	169
COMMAND\ifl@d@esl	169
COMMAND\ifl@d@pnum	169
COMMAND\ifl@d@ssub	169

COMMAND\ifl@dend@X	214
COMMAND\ifl@dmemoir	363
COMMAND\ifl@dpaging	370
COMMAND\ifl@dpairing	73, 365
COMMAND\ifl@dprintingpages	371
COMMAND\ifl@dskipnumber	136
COMMAND\ifl@dstartendok	288
COMMAND\ifl@imakeidx	67
COMMAND\ifledRcol	73, 366
COMMAND\ifledRcol@	73, 369
COMMAND\iflemmacommand@	371
COMMAND\ifnoend@	203
COMMAND\ifnoedgroup@	255
COMMAND\ifnoteschanged@	89
COMMAND\ifnumberedpar@	124
COMMAND\ifnumbering	74, 77
COMMAND\ifnumberingR	74, 366
COMMAND\ifnumberline	115, 136
COMMAND\ifpst@rted	365
COMMAND\ifpst@rtedL	75
COMMAND\ifseriesbefore	217
COMMAND\ifsublines@	86, 98
COMMAND\iftrue	373
COMMAND\ifvmode	233
COMMAND\ifxxx	296
COMMAND\ignorespaces	113
COMMAND\imki@wrindexentry	67
COMMAND\immediate	104, 199
COMMAND\indent	17, 127, 319
COMMAND\index	263
COMMAND\indtl@wrindexentry	67
COMMAND\initnumbering@quote	294, 374
COMMAND\initnumbering@reg	365
COMMAND\initnumbering@sectcmd	374
COMMAND\inplaceoflemmaseparator	318
COMMAND\inplaceofnumber	318
COMMAND\insert	140, 145, 148, 179
COMMAND\insert@count	101, 106, 113
COMMAND\insert@countR	113
COMMAND\insertthangingsymbol	368
COMMAND\insertlines@list	88, 101
COMMAND\insertparafootsepX	193
COMMAND\inserts@list	109, 124, 141, 153
COMMAND\interAfootnotelinepenalty	364
COMMAND\interfootnotelinepenalty	364
COMMAND\interlinepenalty	146
COMMAND\interparanoteglue	316
COMMAND\justifying	38
COMMAND\l@advance@parledegroupp@beforenormalnotes	374
COMMAND\l@d@@wrindexhyp	370

COMMAND\l@d@add	117
COMMAND\l@d@end	199, 214
COMMAND\l@d@nums	113, 115, 117, 169
COMMAND\l@d@section	200
COMMAND\l@d@set	97, 107
COMMAND\l@dampcount	279
COMMAND\l@dbfnote	176, 364
COMMAND\l@dcheckstartend	288
COMMAND\l@dchset@num	97
COMMAND\l@dcolcount	279, 280
COMMAND\l@dcollect@@body	273
COMMAND\l@dcollect@body	272
COMMAND\l@dcsnote	369
COMMAND\l@dcsnotetext	131, 248
COMMAND\l@dcsnotetext@l	131, 248, 249
COMMAND\l@dcsnotetext@r	131, 248
COMMAND\l@ddodorextrafeet	227, 363
COMMAND\l@ddoxtrafeet	227, 363
COMMAND\l@demptyd@ta	365
COMMAND\l@dend@close	199
COMMAND\l@dend@open	199
COMMAND\l@dend@stuff	199
COMMAND\l@denvbody	273
COMMAND\l@dfeetbeginmini	364
COMMAND\l@dfeetendmini	364
COMMAND\l@dgetline@margin	365
COMMAND\l@dgetlock@disp	365
COMMAND\l@dgetref@num	237, 238
COMMAND\l@dgetsidenote@margin	245, 365
COMMAND\l@dgobbeloptarg	370
COMMAND\l@dgobblearg	370
COMMAND\l@dgobbleoptarg	277
COMMAND\l@dlabel@parse	237, 238
COMMAND\l@dld@ta	135, 137
COMMAND\l@dlp@rbox	249
COMMAND\l@dlsn@te	365
COMMAND\l@dlsnote	369
COMMAND\l@dmake@labels	233, 234, 244
COMMAND\l@dmake@labelsR	244
COMMAND\l@dnumpstartsL	75, 365
COMMAND\l@dp@rsefootspec	170
COMMAND\l@dpush@begins	273
COMMAND\l@drd@ta	135, 137
COMMAND\l@dref@undefined	237
COMMAND\l@drsn@te	365
COMMAND\l@drsnote	369
COMMAND\l@dtabaddcols	287
COMMAND\l@dtabnoexpands	363
COMMAND\l@dumboxmpfoot	374
COMMAND\l@dunboxmpfoot	365

COMMAND\@dzeropenalties	365, 370
COMMAND\@pb	308
COMMAND\@prev@nopb	307
COMMAND\@prev@pb	307
COMMAND\@reg	314
COMMAND\label	17, 47, 50, 53, 232, 238
COMMAND\label@refs	231
COMMAND\labelstarttrue	17, 366
COMMAND\labelref@list	231, 234
COMMAND\language	154
COMMAND\last@page@num	364
COMMAND\lastbox	128
COMMAND\lastskip	106
COMMAND\leavevmode	17, 127
COMMAND\led@check@nopb	308
COMMAND\led@check@pb	308
COMMAND\led@nopb	307, 308
COMMAND\led@nopbnum	307
COMMAND\led@pb	307, 308
COMMAND\led@pb@macro	307
COMMAND\led@pbnum	307
COMMAND\led@reinit@index@fornote	263
COMMAND\led@set@index@fornote	263
COMMAND\ledRflag	260
COMMAND\ledchapter	316, 367
COMMAND\ledfootinsdim	317
COMMAND\ledinnernote	50, 247, 369
COMMAND\ledinnote	259, 315, 373
COMMAND\ledinnotemark	52, 315, 372
COMMAND\ledleftnote	50, 247
COMMAND\ledlinenum	85, 365
COMMAND\ledllfill	132
COMMAND\ledlsnotesep	51
COMMAND\ledlsnotewidth	50
COMMAND\lednopb	59, 307
COMMAND\lednopbinverse	308
COMMAND\lednopbinversetrue	44, 59
COMMAND\lednopbnum	307
COMMAND\ledouternote	50, 247, 369
COMMAND\ledpb	59, 307, 316
COMMAND\ledpbnum	307
COMMAND\ledpbsetting	59, 307, 375
COMMAND\ledrightnote	50, 247
COMMAND\ledrsnotesep	51
COMMAND\ledrsnotewidth	50
COMMAND\ledsection	316
COMMAND\ledsectnomark	295
COMMAND\ledsectnotoc	295
COMMAND\ledsetnormalparstuff	315, 316, 372
COMMAND\ledsetnormalparstuff@common	194

COMMAND\ledsetnormalparstuffX	315, 316, 372
COMMAND\ledsidenote	50, 247–249
COMMAND\ledsubsection	316
COMMAND\ledsubsubsection	316
COMMAND\ledxxx	369
COMMAND\left	56
COMMAND\leftctab	289
COMMAND\leftheadline	85
COMMAND\leftlinenum	20, 85, 363, 365
COMMAND\leftltab	288
COMMAND\leftnoteupfalse	50
COMMAND\leftpstartnum	139
COMMAND\leftrtab	289
COMMAND\leftsidenote	248
COMMAND\leftskip	150, 153, 154
COMMAND\lemma	2, 23–25, 27, 28, 109, 112–114, 116, 117, 119, 312, 365, 366, 373, 374, 376
COMMAND\lemmaseparator	318
COMMAND\let	25, 41, 270, 363
COMMAND\letboxXendendlinenum	318
COMMAND\line	159, 162
COMMAND\line@list	87, 102, 115
COMMAND\line@list@stuff	75, 90, 91, 104, 363, 365
COMMAND\line@list@version	93
COMMAND\line@margin	81, 137, 245
COMMAND\line@num	86–88, 135, 363
COMMAND\line@set	117
COMMAND\lineation	19, 80
COMMAND\linebreak	31
COMMAND\linenum	23, 25, 46–48, 109, 117, 236, 238, 244, 312, 377
COMMAND\linenum@out	103, 104, 231, 234
COMMAND\linenumberlist	19, 73, 136, 137, 363
COMMAND\linenumberstyle	21, 84, 363
COMMAND\linenumincrement	19, 365
COMMAND\linenummargin	19, 81, 245
COMMAND\linenumr@p	84, 363, 365
COMMAND\linenumrep	84, 365
COMMAND\linenumsep	20, 51, 85, 246
COMMAND\linerangesep@	225
COMMAND\lineref	231, 236, 244, 318, 371
COMMAND\list@clear	78
COMMAND\list@clearing@reg	365
COMMAND\list@create	78
COMMAND\lock@disp	83
COMMAND\lock@off	99
COMMAND\lock@on	99
COMMAND\lockdisp	20, 83
COMMAND\loop	143, 266
COMMAND\ltab	288, 289, 294
COMMAND\ltabtext	294
COMMAND\m@mmf@prepare	176



COMMAND\makeatletter	131
COMMAND\makeboxoffboxes	155, 157
COMMAND\makeindex	51, 260
COMMAND\makelabel	244
COMMAND\managestanza@modulo	267
COMMAND\marginpar	50, 60, 245, 364
COMMAND\marginparwidth	50, 246
COMMAND\markboth	131
COMMAND\mathchardef	267
COMMAND\maxXnotes	318
COMMAND\maxnotesX	40, 61, 317, 368, 369, 371, 372
COMMAND\maxlinesinpar@list	90
COMMAND\measurebody	290
COMMAND\measuretbody	291
COMMAND\memorybreak	18
COMMAND\morenoexpands	61, 62, 109, 111
COMMAND\morethantwolines	318
COMMAND\mpfnpos	196, 368
COMMAND\mpnnormalfootgroup	364
COMMAND\mpnnormalvfootnote	364
COMMAND\multfootsep	29, 175
COMMAND\multiplefootnotemarker	175
COMMAND\musixtex	369
COMMAND\n@num	365, 372
COMMAND\n@num@ref	372
COMMAND\new@line	105, 364
COMMAND\newcommand	25, 63, 175, 234
COMMAND\newcommandx	25
COMMAND\newhookarg@specific	225
COMMAND\newhookcommand@series	220, 221, 372
COMMAND\newhookcommand@series@reload	221
COMMAND\newhookcommand@toggle@reload	221, 370
COMMAND\newhooktoggle@series	220, 221, 372
COMMAND\newhooktoggle@specific	225
COMMAND\newif	372
COMMAND\newline	31
COMMAND\newlinechar	214
COMMAND\newseries	29, 313, 316, 317
COMMAND\newseries@	207, 218
COMMAND\newverse	44, 45, 316, 369
COMMAND\next	266
COMMAND\next@action	91
COMMAND\next@actionline	91
COMMAND\next@insert	141
COMMAND\nl@regR	93
COMMAND\no@expands	61, 116, 363
COMMAND\noXlemmaseparator	318
COMMAND\nobreak	169
COMMAND\nocritical	208
COMMAND\noedsec	58, 316

COMMAND\noendnotes	316
COMMAND\noexpand	314
COMMAND\nofamiliar	223
COMMAND\noindent	17, 127, 375
COMMAND\noindentX	375
COMMAND\nomk@	225
COMMAND\nonbreakableafternumber	318
COMMAND\nonnumberinfootnote	318
COMMAND\normal@footnotemarkX	179
COMMAND\normal@page@break	307
COMMAND\normal@pars	194
COMMAND\normalbfnoteX	365
COMMAND\normalbodyfootmarkX	179
COMMAND\normalfootfmt	41, 149, 156, 165, 200
COMMAND\normalfootfmtX	180
COMMAND\normalfootfootmarkX	180
COMMAND\normalfootgroup	150
COMMAND\normalfootgroupX	181
COMMAND\normalfootnoterule	146
COMMAND\normalfootstart	149, 153
COMMAND\normalfootstartX	181
COMMAND\normalvfootnote	148
COMMAND\normalvfootnoteX	179
COMMAND\notbool	296
COMMAND\notefontsetup	317
COMMAND\notefontsizeX	37, 317
COMMAND\notenumfont	317
COMMAND\notenumfontX	36, 317
COMMAND\notesXwidthliketwocolumns	318
COMMAND\noteswidthliketwocolumnsX	40, 318, 370, 372
COMMAND\num@lines	124, 142
COMMAND\numberlinefalse	18
COMMAND\numberlinetrue	18
COMMAND\numberonlyfirstinline	218, 318
COMMAND\numberonlyfirstintwolines	318
COMMAND\numberpstartfalse	17
COMMAND\numberpstarttrue	17, 32, 315, 365, 374
COMMAND\numberstanza	33
COMMAND\numberstanzafalse	44
COMMAND\numberstanzatrue	44
COMMAND\numlabfont	20, 41, 85
COMMAND\none@line	124
COMMAND\nonehalfspacing	375
COMMAND\nonlyXpstart	318
COMMAND\npage@action	88, 97
COMMAND\npage@start	88, 365
COMMAND\npagecontents	88
COMMAND\npagelinesep	52
COMMAND\npageno	226
COMMAND\npageparbreak	316

COMMAND\pageref	47, 236
COMMAND\par	24, 31, 127, 128, 194
COMMAND\par@line	124, 142
COMMAND\para@footgroup	153
COMMAND\para@footgroupX	192
COMMAND\para@footsetup	152, 363
COMMAND\para@footsetupX	189, 363, 370
COMMAND\para@vfootnote	157
COMMAND\para@vfootnoteX	190
COMMAND\parafootfmt	155, 156
COMMAND\parafootfmtX	191
COMMAND\parafootftm	158
COMMAND\parafootftmX	193
COMMAND\parafootftmsep	317
COMMAND\parafootsep	318, 368, 373
COMMAND\parafootsepX	38, 90, 317, 318
COMMAND\parafootstart	153
COMMAND\parafootstartX	190
COMMAND\paravfootnote	153
COMMAND\parfillskip	156
COMMAND\parindent	375
COMMAND\parindentX	37, 320, 375, 377
COMMAND\parshape	60
COMMAND\parskip	128
COMMAND\pausenumbering	18, 77, 91, 92, 128, 368, 370, 377
COMMAND\penalty	156
COMMAND\pend	2, 6, 16–19, 21, 58, 107, 109, 112, 118, 124–129, 139, 140, 314, 368, 369
COMMAND\preXnotes	39, 197, 372
COMMAND\preXnotes@	150, 197, 366
COMMAND\prenotesX	39, 198, 372
COMMAND\prepare@preXnotes	196
COMMAND\prev@nopb	307
COMMAND\prev@pb	307
COMMAND\prevlineX	90
COMMAND\prevpageX@num	90
COMMAND\print@Xfootnoterule	373
COMMAND\print@Xnotes	227, 228
COMMAND\print@Xnotes@forpages	371
COMMAND\print@eledsection	131
COMMAND\print@footnoteXrule	373
COMMAND\print@leftmargin@eledsection	297
COMMAND\print@line	129
COMMAND\print@notesX@forpages	371
COMMAND\print@rightmargin@eledsection	297
COMMAND\printendlines	203, 241, 320, 363, 365
COMMAND\printlinefootnote	166, 167, 371
COMMAND\printlinefootnoteara	168, 169, 371
COMMAND\printlinefootnotenumbers	166
COMMAND\printlines	149, 165, 169, 170, 203, 241, 320, 363, 365, 372, 376
COMMAND\printnnum	315, 320

COMMAND\printpstart	166
COMMAND\protect	111, 314
COMMAND\providecommand	175, 363
COMMAND\pstart	2, 6, 16–19, 21, 57, 58, 97, 107, 112, 118, 124, 126–128, 131, 140, 314, 365, 366, 368–370, 372–374, 377
COMMAND\pstartinfootnote	318
COMMAND\pstartinfootnoteeverytime	318
COMMAND\pstartnum	139
COMMAND\pstartref	46, 231, 237, 368
COMMAND\pstarts	366
COMMAND\raggedX	38
COMMAND\raggedleft	38
COMMAND\raggedright	38
COMMAND\raw@text	124
COMMAND\rbracket	35, 36, 41
COMMAND\read@linelist	90–92
COMMAND\ref	47, 50, 53
COMMAND\reformatted@	241
COMMAND\reformattedwithpage	241
COMMAND\relax	17, 97, 133, 141, 270, 277, 314
COMMAND\renewcommand	61, 317, 320
COMMAND\resetprevline@	90
COMMAND\resetprevpage@	90
COMMAND\resumenumbering	18, 74, 77, 91, 92, 128, 365, 369, 370, 377
COMMAND\right	56
COMMAND\rightctab	289
COMMAND\rightlinenum	20, 85, 363, 365
COMMAND\rightltab	289
COMMAND\rightnoteupfalse	50
COMMAND\rightrtab	290
COMMAND\rightsidenote	248
COMMAND\rightright	150, 153, 154, 156
COMMAND\rightstartnum	139
COMMAND\rigidbalance	158–162, 320, 376
COMMAND\rigidbalanceX	159, 320, 376
COMMAND\robustify	31
COMMAND\roman	282, 376
COMMAND\rtab	289, 290, 294
COMMAND\rtabtext	291, 294
COMMAND\sameword	26–28, 118–120, 122, 371, 373, 375
COMMAND\sameword@inedtext	119
COMMAND\saweword	118, 119
COMMAND\scriptsize	85
COMMAND\section	57, 365
COMMAND\section@num	74
COMMAND\sectionmark	295
COMMAND\select@lemmafонт	41, 144
COMMAND\series	207
COMMAND\series@	207
COMMAND\seriesatbegin	29, 217, 372

COMMAND\seriesatend	29, 217, 373
COMMAND\set@line	115
COMMAND\set@line@action	89, 98
COMMAND\setSErefonlypageprefixmore	48, 241, 376
COMMAND\setSErefonlypageprefixsingle	48, 241, 376
COMMAND\setSErefprefixmore	48
COMMAND\setSErefprefixsingle	48
COMMAND\setapprefprefixmore	48, 317
COMMAND\setapprefprefixsingle	48, 317, 376
COMMAND\setcommand@series	219
COMMAND\sethangingsymbol	43, 265, 317, 375
COMMAND\sethangingsymbol	42
COMMAND\setistwofollowinglines	172
COMMAND\setl@dlprbox	249
COMMAND\setline	21, 89, 93, 96, 107, 111, 126, 374
COMMAND\setlinenum	21, 93, 97, 107, 363
COMMAND\setprintendlines	203, 205, 365
COMMAND\setprintlines	170, 172, 203, 365
COMMAND\setsidenoteseq	51
COMMAND\setsidenotsep	317
COMMAND\setstanzaindent	268
COMMAND\setstanzaindents	43, 267, 314
COMMAND\setstanzapenalties	267
COMMAND\setstanzavalues	267
COMMAND\settoggle@series	218, 366, 370
COMMAND\showlemma	110, 364
COMMAND\showwordrank	28, 119
COMMAND\sidenote@margin	364
COMMAND\sidenotemargin	50, 364, 369
COMMAND\sidenoteseq	317
COMMAND\sidepstartnumtrue	17
COMMAND\skip	149, 150
COMMAND\skipnumbering	21, 100, 108, 365, 373
COMMAND\skipnumbering@reg	373
COMMAND\small	37
COMMAND\special	12
COMMAND\splitmaxdepth	146, 160
COMMAND\splitoff	159
COMMAND\splittopskip	146, 160, 161
COMMAND\stanza	20, 21, 44, 45, 270, 317, 319, 375
COMMAND\stanza@hang	270
COMMAND\stanza@line	270
COMMAND\stanzaindent	43, 268, 371
COMMAND\stanzaindent*	43
COMMAND\stanzaindentbase	267
COMMAND\stanzanumwrapper	44
COMMAND\startlock	20, 89, 108, 270
COMMAND\startstanzahook	317
COMMAND\startsub	20, 88, 106
COMMAND\strip@pt	153

COMMAND\strutbox	160
COMMAND\sub@action	88, 98
COMMAND\sub@lock	87
COMMAND\sub@off	95, 234
COMMAND\sub@on	95, 234
COMMAND\subline@num	86–88
COMMAND\sublinenum@rep	363
COMMAND\sublinenumberstyle	21, 84, 363
COMMAND\sublinenumincrement	19
COMMAND\sublinenumr@p	84, 363, 365
COMMAND\sublinenumrep	84, 365
COMMAND\sublineref	46, 231, 237
COMMAND\subsectionmark	295
COMMAND\sw@inthisedtext	112
COMMAND\sw@list@inedtext	116, 122
COMMAND\symlinenum	318
COMMAND\symplinum	317
COMMAND\sza@penalty	270
COMMAND>tag	371
COMMAND\text	312
COMMAND\textcolor	62
COMMAND\textheight	61
COMMAND\the	363
COMMAND\thefootnoteA	28
COMMAND\thefootnoteX	367
COMMAND\thelabidx	262
COMMAND\thepage	93
COMMAND\thepstart	17
COMMAND\thepstartL	366
COMMAND\thepstartR	366
COMMAND\thestanza	44
COMMAND>this@line@list@version	104
COMMAND>thisfootnote	183
COMMAND\threecolfootfmt	161
COMMAND\threecolfootfmtX	188
COMMAND\threecolfootgroup	160
COMMAND\threecolfootgroupX	188
COMMAND\threecolfootsetup	160
COMMAND\threecolfootsetupX	187
COMMAND\threecolvfootnote	160
COMMAND\threecolvfootnoteX	187
COMMAND\twocolfootfmtX	186
COMMAND\twocolfootgroupX	186
COMMAND\twocolfootsetupX	185
COMMAND\twocolvfootnoteX	185
COMMAND\twolines	218, 318
COMMAND\twolines@A	218
COMMAND\twolines@B	218
COMMAND\twolines@C	218
COMMAND\twolinesbutnotmore	318

COMMAND\twolinesonlyinsamepage	318
COMMAND\txbeforeXnotes	318
COMMAND\unhbox	154
COMMAND\unpenalty	155–157
COMMAND\unskip	156
COMMAND\unvxh	156, 319
COMMAND\unvxhX	319
COMMAND\upbracefill	288
COMMAND\usingcritext	313, 316
COMMAND\usingedtext	313, 316
COMMAND\vAfootnote	146
COMMAND\variant	25
COMMAND\vbox	125, 127, 154, 158, 159, 196
COMMAND\vfootnote	145, 150, 153, 160
COMMAND\vl@dbfnote	176, 364
COMMAND\vnumfootnoteX	365
COMMAND\vsizex	39, 40, 61
COMMAND\vsplit	142
COMMAND\waklam	288
COMMAND\waklamec	288
COMMAND\wapunktel	288
COMMAND\wastricht	288
COMMAND\widthX	40, 320, 377
COMMAND\wrap@edcrossref	236, 370
COMMAND\wrapped@bodyfootmarkX	193
COMMAND\wrapped@footfootmarkX	193
COMMAND\x...	46
COMMAND\xdef	79, 270
COMMAND\xflagref	47, 237, 320, 376
COMMAND\xleft@appenditem	79, 110
COMMAND\xlineref	46, 47, 320, 376
COMMAND\xpageref	46
COMMAND\xpstartref	46, 368
COMMAND\xr	50
COMMAND\xright@appenditem	79
COMMAND\xsublineref	46
COMMAND\xxref	47, 238, 244, 368, 371, 372
COMMAND\zz@@@	363
ENVIRONMENTastanza	376
ENVIRONMENTedarrayc	294
ENVIRONMENTedarrayl	294
ENVIRONMENTedarrayr	294
ENVIRONMENTedtabularc	294
ENVIRONMENTedtabularl	294
ENVIRONMENTedtabularr	294
ENVIRONMENTledgroup	66, 254, 320, 376
ENVIRONMENTledgroupsize	254
PACKAGE(r)(e)ledmac	29
PACKAGEEledmac	11, 63, 88, 259, 315, 316, 371–373
PACKAGEEledpar	372, 373

PACKAGEetoolbox	65
PACKAGEParallel	321
PACKAGEReledmac	319, 320
PACKAGEamsgen	272
PACKAGEamsmath	272
PACKAGEbabel	62, 282, 376
PACKAGEbiblatex	60
PACKAGEbidi	67, 375
PACKAGEcaption	73
PACKAGEcolor	61
PACKAGEedmac	1, 5, 10–13, 63, 169, 175, 232, 267, 312, 321, 363
PACKAGEedstanza	1, 13, 265
PACKAGEeledmac	1, 10, 13–15, 52, 118, 175, 256, 259, 275, 298, 310, 313, 315–317, 367, 369, 371
PACKAGEeledpar	73, 146, 296, 321, 366, 369–372
PACKAGEetex	375
PACKAGEetoolbox	78, 118, 207, 218, 226, 248, 296, 307
PACKAGEfloatrow	60, 320
PACKAGEfootmisc	29, 62, 175, 321
PACKAGEglossaries	53, 264, 376
PACKAGEhandout	370
PACKAGEhyperlink	214
PACKAGEhyperref	47, 112, 193, 232, 261, 300, 309, 368–370, 377
PACKAGEifluatex	65
PACKAGEifxetex	65
PACKAGEimakeidx	51, 60, 66, 67, 256, 259, 316, 367–369, 371
PACKAGEindextols	263
PACKAGEindextool	316
PACKAGEindextools	51, 60, 66, 67, 74, 256, 259, 263, 316, 371, 376
PACKAGEinputenc	120
PACKAGEledarab	62
PACKAGEledmac	1, 10, 13, 62, 78, 259, 312, 313, 316, 319
PACKAGEledpar	62
PACKAGEMemoir	66, 259, 316, 321, 370
PACKAGEMorewrites	60
PACKAGEMusixtex	369
PACKAGEperpage	376
PACKAGEpolyglossia	35, 62, 113, 145, 165, 376
PACKAGERagged2e	38, 65
PACKAGEReledmac	1, 2, 10, 11, 13–16, 18, 19, 21–23, 25–27, 29, 31, 34, 37–39, 41, 43, 45–48, 50–54, 58, 60–64, 80, 82, 88, 89, 92, 93, 95, 103, 104, 111, 140, 147, 150, 154, 175, 200, 208, 211, 212, 218, 226, 236, 239, 244, 259, 275, 296, 297, 308, 309, 316, 317, 319, 320, 374, 377
PACKAGEReledpar	1, 4, 5, 8, 14, 17, 40, 47, 49, 58–60, 62, 64, 73, 80, 90, 95, 113, 148, 150, 194, 195, 207, 213, 225, 227, 228, 255, 265, 375, 376
PACKAGESuffix	65
PACKAGETabmac	1, 13, 321
PACKAGEuninormalize	26
PACKAGExargs	25, 65
PACKAGExkeyval	64, 225
PACKAGEXI	4, 50, 244, 377
PACKAGEXref	244



PACKAGEstring . . . . . 65, 261

## A

\absline@num . . . . . 1  
 Abu Kamil Shuja' b. Aslam . . . . . 12  
 \actionlines@list . . . . . 1  
 \actions@list . . . . . 1  
 \add@inserts . . . . . 1  
 \add@inserts@next . . . . . 1  
 \add@penalties . . . . . 1  
 \addtol@denvbody . . . . . 1  
 Adelard II . . . . . 12  
 \advancelabel@refs . . . . . 1  
 \advanceline . . . . . 1, 20  
 \advancepageno . . . . . 1  
 \Aendnote . . . . . 23  
 \affixline@num . . . . . 1  
 \affixpstart@num . . . . . 1  
 \affixside@note . . . . . 1  
 \Afootnote . . . . . 23  
 \afternoteX . . . . . 38  
 \afterruleX . . . . . 39  
 \ampersand . . . . . 1, 44  
 \applabel . . . . . 1, 48  
 \appref . . . . . 1, 48  
 \apprefwithpage . . . . . 1, 48  
 \arrangementX . . . . . 1, 30  
 \arrangementX@normal . . . . . 1  
 \arrangementX@threecol . . . . . 1  
 \arrangementX@twocol . . . . . 1  
 \at@every@pend . . . . . 1  
 \AtEveryPend . . . . . 1, 17  
 \AtEveryPstart . . . . . 1, 17  
 \autopar . . . . . 1, 16

## B

\ballast . . . . . 60  
 \ballast@count . . . . . 1  
 Beeton, Barbara Ann Neuhaus Friend . . . . . 17  
 \beforeeledchapter . . . . . 1  
 \beforenotesX . . . . . 39  
 \beginnumbering . . . . . 1, 15  
 \Bendnote . . . . . 23  
 \Bfootnote . . . . . 23  
 \bhookgroupX . . . . . 39  
 \bhooknoteX . . . . . 37  
 \bodyfootmarkA . . . . . 28  
 \boxfootnotenumbers . . . . . 1  
 Bredon, Simon . . . . . 12  
 Breger, Herbert . . . . . 12, 13, 275

Brey, Gerhard .....	12
Busard, Hubert L. L. ....	12
\bypage@false .....	1
\bypage@true .....	1
\bypstart@false .....	1
\bypstart@true .....	1

## C

\c@addcolcount .....	1
\c@ballast .....	1
\c@firstlinenum .....	1
\c@firstsublinenum .....	1
\c@labidx .....	1
\c@linenumincrement .....	1
\c@sublinenumincrement .....	1
\Cendnote .....	23
\Cfootnote .....	23
\ch@ck@l@ck .....	1
\ch@cksub@l@ck .....	1
\chapter .....	1
\check@pb@in@verse .....	1
Chester, Robert of .....	12
Claassens, Geert H. M. ....	12
\colalignX .....	37
Copernicus, Nicolaus .....	12
\critext .....	312
\ctab .....	1
\ctabtext .....	1

## D

Dekker, Dirk-Jan .....	61
\Dendnote .....	23
\Dfootnote .....	23
\disable@familiarnotes .....	1
\disable@notes .....	1
\disable@sidenotes .....	1
\disablel@dtabfeet .....	1
\do@actions .....	1
\do@actions@fixedcode .....	1
\do@actions@next .....	1
\do@ballast .....	1
\do@feetX .....	1
\do@insidelinehook .....	1
\do@line .....	1
\do@linehook .....	1
\do@lockoff .....	1
\do@lockoffL .....	1
\do@lockon .....	1
\do@lockonL .....	1
\doedindexlabel .....	1

<code>\doendnotes</code>	1, 24
<code>\doendnotesbysection</code>	1, 24
<code>\doinsidelinehook</code>	1, 21
<code>\dolinehook</code>	1, 21
<code>\dosplits</code>	1
Downes, Michael	61, 154, 156
<code>\doxtrafeet</code>	1
<code>\dummy@edtext</code>	1
<code>\dummy@edtext@showlemma</code>	1
<code>\dummy@ref</code>	1

## E

<code>\edaftertab</code>	1, 56, 288
<code>edarrayc</code> (environment)	54
<code>edarrayl</code> (environment)	54
<code>edarrayr</code> (environment)	54
<code>\edatleft</code>	1, 56
<code>\edatright</code>	1, 56
<code>\edbeforetab</code>	1, 56, 288
<code>\edfilldimen</code>	1
<code>\edfont@info</code>	1
<code>\EDINDEX</code>	1
<code>\edindex</code>	1, 51
<code>\edindexlab</code>	1, 53
<code>\EDLABEL</code>	1
<code>\edlabel</code>	1, 45
<code>\edlabelE</code>	1, 47
<code>\edlabelS</code>	1, 47
<code>\edlabelSE</code>	1, 47
<code>\edlineref</code>	1, 46
<code>\edmakelabel</code>	1, 47
<code>\edpageref</code>	1, 46
<code>\edrowfill</code>	1, 55
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 54
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1
<code>\EDTABtext</code>	1
<code>edtabularc</code> (environment)	54
<code>edtabularl</code> (environment)	54
<code>edtabularr</code> (environment)	54
<code>\EDTEXT</code>	1
<code>\edtext</code>	1, 22
<code>\edvertdots</code>	1, 56
<code>\edvertline</code>	1, 56
<code>\Eendnote</code>	23
<code>\Efootnote</code>	23
<code>\eled@chapter</code>	1
<code>\eled@section</code>	1
<code>\eled@sectioning@out</code>	1

<code>\eled@subsection</code>	1
<code>\eled@subsubsection</code>	1
<code>\eledchapter</code>	1
<code>\eledchapter*</code>	1
<code>\eledsection</code>	1
<code>\eledsection*</code>	1
<code>\eledsubsection</code>	1
<code>\eledsubsection*</code>	1
<code>\eledsubsubsection</code>	1
<code>\eledsubsubsection*</code>	1
<code>\enablel@dtabfeet</code>	1
<code>\end@lemmas</code>	1
<code>\endashchar</code>	1
<code>\endline@num</code>	1
<code>\endlock</code>	1, 20
<code>\endminipage</code>	1
<code>\endnumbering</code>	1, 15
<code>\endpage@num</code>	1
<code>\endprint</code>	1
<code>\endquotation</code>	1
<code>\endquote</code>	1
<code>\endsub</code>	1, 20
<code>\endsubline@num</code>	1
environments:	
<code>edarrayc</code>	54
<code>edarrayl</code>	54
<code>edarrayr</code>	54
<code>edtabularc</code>	54
<code>edtabularl</code>	54
<code>edtabularr</code>	54
<code>ledgroup</code>	45
<code>ledgroupsized</code>	45
<code>minipage</code>	45
Euclid	12
<code>\extensionchars</code>	1, 59

## F

<code>\f@x@l@cks</code>	1
Fairbairns, Robin	29
<code>\first@linenum@out@false</code>	1
<code>\first@linenum@out@true</code>	1
<code>\firstlinenum</code>	1, 19
<code>\firstseriesX@</code>	1
<code>\firstsublinenum</code>	1, 19
<code>\firstXseries@</code>	1
<code>\fix@page</code>	1
<code>\flag@end</code>	1
<code>\flag@start</code>	1
<code>\flagstanza</code>	1, 44
<code>\flush@notes</code>	1

<code>\fnpos</code> .....	1, 29
Folkerts, Menso .....	12
<code>\footfootmarkA</code> .....	28
<code>\footfudgefiddle</code> .....	1, 61
<code>\footnote</code> .....	1
<code>\footnoteA</code> .....	28
<code>\footnoteB</code> .....	28
<code>\footnoteC</code> .....	28
<code>\footnoteD</code> .....	28
<code>\footnoteE</code> .....	28
<code>\footnotelang@lua</code> .....	1
<code>\footnotelang@poly</code> .....	1
<code>\footnoteoptions@</code> .....	1
<code>\footsplitskips</code> .....	1
<code>\fullstop</code> .....	1

## G

Gädeke, Nora .....	12
<code>\get@edindex@hyperref</code> .....	1
<code>\get@edindex@ledinnote@command</code> .....	1
<code>\get@fnmark</code> .....	1
<code>\get@fnmarkX</code> .....	1
<code>\get@index@command</code> .....	1
<code>\get@linelistfile</code> .....	1
<code>\get@sw@txt</code> .....	1
<code>\get@thisfootnote</code> .....	1
<code>\get@thisfootnoteX</code> .....	1
<code>\getline@num</code> .....	1
<code>\gl@p</code> .....	1

## H

<code>\h@num</code> .....	1
<code>\hangindentX</code> .....	37
<code>\hidenumbering</code> .....	1, 21
<code>\Hilfsbox</code> .....	1
<code>\hilfsbox</code> .....	1
<code>\hilfscount</code> .....	1
<code>\HILFSskip</code> .....	1
<code>\Hilfsskip</code> .....	1
<code>\hilfsskip</code> .....	1
<code>\hsizethreecolX</code> .....	38
<code>\hsizetwocolX</code> .....	38
<code>\Hy@raisedlink@left</code> .....	1
<code>\hyperlinkformat</code> .....	1
<code>\hyperlinkformatR</code> .....	1
<code>\hyperlinkR</code> .....	1

## I

<code>\if@addsw</code> .....	1
<code>\if@edindex@fornote@true</code> .....	1

<code>\ifeled@sectioning</code>	1
<code>\ifled@nofoot</code>	1
<code>\ifledgroup</code>	1
<code>\iflemmacommand@</code>	1
<code>\ifnoeled@sec</code>	1
<code>\ifnoneed@Footnote</code>	1
<code>\if@RTL</code>	1
<code>\ifautopar@pause</code>	1
<code>\ifbypage@</code>	1
<code>\ifbypage@R</code>	1
<code>\ifbypstart@</code>	1
<code>\ifbypstart@R</code>	1
<code>\ifeledmaccompat@</code>	1
<code>\iffirst@linenum@out@</code>	1
<code>\ifindtl@innote</code>	1
<code>\ifindtl@notenumber</code>	1
<code>\ifinserthangingsymbol</code>	1
<code>\ifinstanza</code>	1
<code>\ifl@d@dash</code>	1
<code>\ifl@d@elin</code>	1
<code>\ifl@d@esl</code>	1
<code>\ifl@d@pnum</code>	1
<code>\ifl@d@ssub</code>	1
<code>\ifl@d@Xmorethantwolines</code>	1
<code>\ifl@d@Xtwolines</code>	1
<code>\ifl@dend@X</code>	1
<code>\ifl@dhiddenumber</code>	1
<code>\ifl@dmemoir</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dprintingcolumns</code>	1
<code>\ifl@dprintingpages</code>	1
<code>\ifl@dskipnumber</code>	1
<code>\ifl@dskipversenumber</code>	1
<code>\ifl@dstartendok</code>	1
<code>\ifl@imakeidx</code>	1
<code>\ifl@indextools</code>	1
<code>\ifledfinal</code>	1, 59
<code>\ifledgroupnotesL@</code>	1
<code>\ifledgroupnotesR@</code>	1
<code>\iflednopbinverse</code>	1
<code>\ifledRcol</code>	1
<code>\ifledRcol@</code>	1
<code>\ifnocritical@</code>	1
<code>\ifnoend@</code>	1
<code>\ifnofamiliar@</code>	1
<code>\ifnoledgroup@</code>	1
<code>\ifnoquotation@</code>	1
<code>\ifnoteschanged@</code>	1
<code>\ifnumberedpar@</code>	1

<code>\ifnumbering</code> .....	1
<code>\ifnumberingR</code> .....	1
<code>\ifnumberline</code> .....	1
<code>\ifnumberstanza</code> .....	1
<code>\ifparapparatus@</code> .....	1
<code>\ifparledgroup</code> .....	1
<code>\ifpst@rtedL</code> .....	1
<code>\ifseriesbefore</code> .....	1
<code>\ifsidepstartnum</code> .....	1
<code>\ifsublines@</code> .....	1
<code>\ifwidthliketwocolumns</code> .....	1
<code>\ifXendinsertsep@</code> .....	1
<code>\ifxindy@</code> .....	1
<code>\ifxindyhyperref@</code> .....	1
<code>\initnumbering@quote</code> .....	1
<code>\initnumbering@reg</code> .....	1
<code>\insert@count</code> .....	0, 1
<code>\inserthangingymbol</code> .....	1
<code>\insertlines@list</code> .....	1
<code>\inserts@list</code> .....	1

## J

Jayaditya .....	12
-----------------	----

## K

Kabelschacht, Alois .....	143
---------------------------	-----

## L

<code>\l@advance@parledgroup@beforenormalnotes</code> .....	1
<code>\l@d@add</code> .....	1
<code>\l@d@nums</code> .....	1
<code>\l@d@section</code> .....	1
<code>\l@d@set</code> .....	1
<code>\l@d@Xend</code> .....	1
<code>\l@dampcount</code> .....	1
<code>\l@dbfnote</code> .....	1
<code>\l@dcheckcols</code> .....	1
<code>\l@dcheckstartend</code> .....	1
<code>\l@dchset@num</code> .....	1
<code>\l@dcolcount</code> .....	1
<code>\l@dcollect@body</code> .....	1
<code>\l@dcollect@body</code> .....	1
<code>\l@dcolwidth</code> .....	1
<code>\l@dcsnote</code> .....	1
<code>\l@dcsnotetext</code> .....	1
<code>\l@dcsnotetext@l</code> .....	1
<code>\l@dcsnotetext@r</code> .....	1
<code>\l@ddodoreintrafeet</code> .....	1
<code>\l@dedbeginmini</code> .....	1
<code>\l@dedendmini</code> .....	1

<code>\@emptyd@ta</code>	1
<code>\@dend@close</code>	1
<code>\@dend@open</code>	1
<code>\@dend@stuff</code>	1
<code>\@dend@Xfalse</code>	1
<code>\@dend@Xtrue</code>	1
<code>\@denvbody</code>	1
<code>\@dfambeginmini</code>	1
<code>\@dfamendmini</code>	1
<code>\@dfeetbeginmini</code>	1
<code>\@dfeetendmini</code>	1
<code>\@dgetline@margin</code>	1
<code>\@dgetlock@disp</code>	1
<code>\@dgetref@num</code>	1
<code>\@dgetsidenote@margin</code>	1
<code>\@dgobblearg</code>	1
<code>\@dlabel@parse</code>	1
<code>\@dld@ta</code>	1
<code>\@dlp@rbox</code>	1
<code>\@dlsn@te</code>	1
<code>\@dlsnote</code>	1
<code>\@dmake@labels</code>	1
<code>\@dmodforedtext</code>	1
<code>\@dnullfills</code>	1
<code>\@dnumpstartsL</code>	1
<code>\@doldold@footnotetext</code>	1
<code>\@dp@rsefootspec</code>	1
<code>\@dparsedendline</code>	1
<code>\@dparsedendpage</code>	1
<code>\@dparsedendsub</code>	1
<code>\@dparsedstartline</code>	1
<code>\@dparsedstartpage</code>	1
<code>\@dparsedstartsub</code>	1
<code>\@dpush@begins</code>	1
<code>\@drd@ta</code>	1
<code>\@dref@undefined</code>	1
<code>\@drestorefills</code>	1
<code>\@drestoreforedtext</code>	1
<code>\@drp@rbox</code>	1
<code>\@drsn@te</code>	1
<code>\@drsnote</code>	1
<code>\@dsetmaxcolwidth</code>	1
<code>\@dskipnumberfalse</code>	1
<code>\@dskipnumbertrue</code>	1
<code>\@dtabaddcols</code>	1
<code>\@dtabnoexpands</code>	1
<code>\@dunboxmpfoot</code>	1
<code>\@dunhbox@line</code>	1
<code>\@dzeropenalties</code>	1
<code>\label</code>	47



<code>\labelstartfalse</code> .....	1
<code>\labelstarttrue</code> .....	1, 17
<code>\labelref@list</code> .....	1
<code>\labelrefsparseline</code> .....	1
<code>\labelrefsparsesubline</code> .....	1
<code>\last@page@num</code> .....	1
Lavagnino, John .....	11
<code>\led@check@nopb</code> .....	1
<code>\led@check@pb</code> .....	1
<code>\led@err@AutoparNotNumbered</code> .....	1
<code>\led@err@edtextoutsidepstart</code> .....	1
<code>\led@err@EdtextWithoutFootnote</code> .....	1
<code>\led@err@FootnoteWithoutEdtext</code> .....	1
<code>\led@err@HighEndColumn</code> .....	1
<code>\led@err@LineationInNumbered</code> .....	1
<code>\led@err@LowStartColumn</code> .....	1
<code>\led@err@ManyLeftnotes</code> .....	1
<code>\led@err@ManyRightnotes</code> .....	1
<code>\led@err@ManySidenotes</code> .....	1
<code>\led@err@NumberingNotStarted</code> .....	1
<code>\led@err@NumberingShouldHaveStarted</code> .....	1
<code>\led@err@NumberingStarted</code> .....	1
<code>\led@err@NumberingWithoutPstart</code> .....	1
<code>\led@err@PendNoPstart</code> .....	1
<code>\led@err@PendNotNumbered</code> .....	1
<code>\led@err@PstartInPstart</code> .....	1
<code>\led@err@PstartNotNumbered</code> .....	1
<code>\led@err@ReverseColumns</code> .....	1
<code>\led@err@TooManyColumns</code> .....	1
<code>\led@err@UnequalColumns</code> .....	1
<code>\led@error@fail@patch@@doclearpage</code> .....	1
<code>\led@error@fail@patch@@iiiminipage</code> .....	1
<code>\led@error@fail@patch@@makecol</code> .....	1
<code>\led@error@fail@patch@@reinserts</code> .....	1
<code>\led@error@fail@patch@@endminipage</code> .....	1
<code>\led@error@ImakeidxAfterEledmac</code> .....	1
<code>\led@error@IndextoolsAfterEledmac</code> .....	1
<code>\led@mess@NotesChanged</code> .....	1
<code>\led@mess@SectionContinued</code> .....	1
<code>\led@nopb</code> .....	1
<code>\led@nopbnum</code> .....	1
<code>\led@pb</code> .....	1
<code>\led@pb@setting</code> .....	1
<code>\led@pbnum</code> .....	1
<code>\led@reinit@index@fornote</code> .....	1
<code>\led@set@index@fornote</code> .....	1
<code>\led@toksa</code> .....	1
<code>\led@toksb</code> .....	1
<code>\led@warn@AppLabelOutEdtext</code> .....	1
<code>\led@warn@BadAction</code> .....	1

<code>\led@warn@BadAdvancelineLine</code>	1
<code>\led@warn@BadAdvancelineSubline</code>	1
<code>\led@warn@BadLineation</code>	1
<code>\led@warn@BadLinenummargin</code>	1
<code>\led@warn@BadLockdisp</code>	1
<code>\led@warn@BadSetline</code>	1
<code>\led@warn@BadSetlinenum</code>	1
<code>\led@warn@BadSidenotemargin</code>	1
<code>\led@warn@BadSublockdisp</code>	1
<code>\led@warn@DuplicateLabel</code>	1
<code>\led@warn@LineFileObsolete</code>	1
<code>\led@warn@NoIndexFile</code>	1
<code>\led@warn@NoLineFile</code>	1
<code>\led@warn@NoMarginpars</code>	1
<code>\led@warn@RefUndefined</code>	1
<code>\led@warn@SeriesStillExist</code>	1
<code>\led@warning@hsizeX@deprecated</code>	1
<code>\led@warning@Xhsize@deprecated</code>	1
<code>ledgroup</code> (environment)	45
<code>ledgroupsized</code> (environment)	45
<code>\ledinnernote</code>	1, 50
<code>\ledinnote</code>	1
<code>\ledinnotehyperpage</code>	1
<code>\ledinnotemark</code>	1
<code>\ledleftnote</code>	1, 50
<code>\ledlinenum</code>	1
<code>\ledllfill</code>	1
<code>\ledlsnotefontsetup</code>	1, 51
<code>\ledlsnotesep</code>	1, 51
<code>\ledlsnotewidth</code>	1, 50
<code>\lednopb</code>	1, 58
<code>\lednopbinversetrue</code>	59
<code>\lednopbnum</code>	1
<code>\ledouternote</code>	50
<code>\ledouterote</code>	1
<code>\ledpb</code>	1, 58
<code>\ledpbnum</code>	1
<code>\ledpbsetting</code>	1, 59
<code>\ledrightnote</code>	1, 50
<code>\ledrlfill</code>	1
<code>\ledrsnotefontsetup</code>	1, 51
<code>\ledrsnotesep</code>	1, 51
<code>\ledrsnotewidth</code>	1, 50
<code>\ledsectnomark</code>	1
<code>\ledsectnotoc</code>	1
<code>\ledsetnormalparstuff@common</code>	1
<code>\ledsetnormalparstuffX</code>	1
<code>\ledsidenote</code>	1, 50
<code>\leftctab</code>	1
<code>\leftlinenum</code>	1, 20

<code>\leftltab</code>	1
<code>\leftnoteupfalse</code>	50
<code>\leftpstartnum</code>	1
<code>\leftrtab</code>	1
Leibniz	12
<code>\lemma</code>	1, 24
<code>\letsforverteilen</code>	1
<code>\line@list</code>	1
<code>\line@list@stuff</code>	1
<code>\line@list@version</code>	1
<code>\line@margin</code>	1
<code>\line@num</code>	1
<code>\line@set</code>	1
<code>\lineation</code>	1, 19
<code>\linenum</code>	1, 25
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 19
<code>\linenumberstyle</code>	1, 21
<code>\linenumincrement</code>	1, 19
<code>\linenummargin</code>	1, 19
<code>\linenumr@p</code>	1
<code>\linenumrep</code>	1
<code>\linenumsep</code>	1, 20
<code>\linerangesep@</code>	1
<code>\list@clear</code>	1
<code>\list@clearing@reg</code>	1
<code>\list@create</code>	1
<code>\lock@disp</code>	1
<code>\lock@off</code>	1
<code>\lock@on</code>	1
<code>\lockdisp</code>	1, 20
Lorch, Richard	12
<code>\ltab</code>	1
<code>\ltabtext</code>	1
Luecking, Dan	65

## M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\managestanza@modulo</code>	1
<code>\maxhnotesX</code>	40
Mayer, Gyula	12
<code>\measuretbody</code>	1
<code>\measuremcell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1

Middleton, Thomas	12, 86
minipage (environment)	45
Mittelbach, Frank	12
\morenoexpands	1, 61
\mpfnpos	1, 29
\mpnormalfootgroup	1
\mpnormalfootgroupX	1
\mpnormalvfootnote	1
\mpnormalvfootnoteX	1
\mppara@footgroupX	1
\mppara@vfootnoteX	1
\mpparafootgroup	1
\mpparavfootnote	1
\mpthreecolfootgroup	1
\mpthreecolfootgroupX	1
\mpthreecolfootsetup	1
\mpthreecolfootsetupX	1
\mptwocolfootgroup	1
\mptwocolfootgroupX	1
\mptwocolfootsetup	1
\mptwocolfootsetupX	1
\multfootsep	1, 29
\multiplefootnotemarker	1

## N

\n@num	1
\n@num@stanza	1
\new@line	1
\newhookarg@specific	1
\newhookcommand@series	1
\newhookcommand@series@reload	1
\newhooktoggle@series	1
\newhooktoggle@series@reload	1
\newhooktoggle@specific	1
\newseries@	1
\newverse	1
\NEXT	1
\no@expands	1
\noeledsec	58
\nomk@	1
\normal@footnotemarkX	1
\normal@page@break	1
\normal@pars	1
\normalbfnoteX	1
\normalbodyfootmarkX	1
\normalfootfmt	1
\normalfootfmtX	1
\normalfootfootmarkX	1
\normalfootgroup	1
\normalfootgroupX	1

<code>\normalfootnoterule</code>	1
<code>\normalfootnoteruleX</code>	1
<code>\normalfootstart</code>	1
<code>\normalfootstartX</code>	1
<code>\normalvfootnote</code>	1
<code>\normalvfootnoteX</code>	1
<code>\notefontsizeX</code>	37
<code>\notenumfontX</code>	36
<code>\noteschanged@false</code>	1
<code>\noteschanged@true</code>	1
<code>\noteswidthliketwocolumnsX</code>	40
<code>\nulledindex</code>	1
<code>\nullsetzen</code>	1
<code>\num@lines</code>	1
<code>\numberedpar@false</code>	1
<code>\numberedpar@true</code>	1
<code>\numberingfalse</code>	1
<code>\numberingtrue</code>	1
<code>\numberlinefalse</code>	18
<code>\numberlinetrue</code>	18
<code>\numberpstartfalse</code>	1, 17
<code>\numberpstarttrue</code>	1, 17
<code>\numberstanzafalse</code>	44
<code>\numberstanzatrue</code>	44
<code>\numlabfont</code>	1, 41

## O

<code>\old@hsize</code>	1
<code>\one@line</code>	1
<code>optioncontinuousnumberingwithcolumns</code>	377
<code>optioninnnote</code>	376
<code>optioninnote</code>	376
<code>optionlinangesep</code>	225
<code>optionnocritical</code>	376
<code>optionnoend</code>	376
<code>optionnotenumber</code>	376

## P

<code>\page@action</code>	1
<code>\page@num</code>	1
<code>\pagelinesep</code>	1, 52
<code>\pageno</code>	1
<code>\pageref</code>	47
<code>\par@line</code>	1
<code>\para@footgroupX</code>	1
<code>\para@footsetup</code>	1
<code>\para@footsetupX</code>	1
<code>\para@vfootnoteX</code>	1
<code>\parafootfmt</code>	1
<code>\parafootfmtX</code>	1

<code>\parafootgroup</code> .....	1
<code>\parafootsepX</code> .....	38
<code>\parafootstart</code> .....	1
<code>\parafootstartX</code> .....	1
<code>\paravfootnote</code> .....	1
<code>\parindentX</code> .....	37
<code>\pausenumbering</code> .....	1, 18
<code>\pend</code> .....	1, 16
Plato of Tivoli .....	12
<code>\postbodyfootmark</code> .....	1
<code>\prebodyfootmark</code> .....	1
<code>\prenotesX</code> .....	39
<code>\prepare@edindex@fornote</code> .....	1
<code>\prepare@prenotesX</code> .....	1
<code>\prepare@preXnotes</code> .....	1
<code>\prev@nopb</code> .....	1
<code>\prev@pb</code> .....	1
<code>\prevpage@num</code> .....	1
<code>\preXnotes</code> .....	1, 39
<code>\preXnotes@</code> .....	1
<code>\print@eledsection</code> .....	1
<code>\print@footnoteXrule</code> .....	1
<code>\print@leftmargin@eledsection</code> .....	1
<code>\print@line</code> .....	1
<code>\print@notesX</code> .....	1
<code>\print@rightmargin@eledsection</code> .....	1
<code>\print@Xfootnoterule</code> .....	1
<code>\print@Xnotes</code> .....	1
<code>\printendlines</code> .....	1
<code>\printlineendnote</code> .....	1
<code>\printlineendnotearea</code> .....	1
<code>\printlinefootnote</code> .....	1
<code>\printlinefootnotearea</code> .....	1
<code>\printlinefootnotenumbers</code> .....	1
<code>\printlines</code> .....	1
<code>\printnpnum</code> .....	1
<code>\printpstart</code> .....	1
<code>\printsymlineendnotearea</code> .....	1
<code>\printsymlinefootnotearea</code> .....	1
<code>\printXafternumber</code> .....	1
<code>\printXbeforenumber</code> .....	1
<code>\pstart</code> .....	1, 16
<code>\pstarteref</code> .....	1
<code>\pstartnum</code> .....	1
<code>\pstartref</code> .....	46
<b>Q</b>	
<code>\quotation</code> .....	1
<code>\quote</code> .....	1

## R

<code>\raggedX</code>	38
<code>\raw@text</code>	1
<code>\rbracket</code>	1
<code>\read@linelist</code>	1
<code>\ref</code>	47
<code>\Relax</code>	1
<code>\reledmac@error</code>	1
<code>\reledmac@warning</code>	1
<code>\removehboxes</code>	1
<code>\resetprevline@</code>	1, 90
<code>\resetprevpage@</code>	1
<code>\resetprevpage@num</code>	90
<code>\restore@familiarnotes</code>	1
<code>\restore@notes</code>	1
<code>\restore@sidenotes</code>	1
<code>\resumenumbering</code>	1, 18
<code>\rightctab</code>	1
<code>\rightlinenum</code>	1, 20
<code>\rightltab</code>	1
<code>\rightnoteupfalse</code>	50
<code>\rightrtab</code>	1
<code>\rightstartnum</code>	1
<code>\rigidbalance</code>	1
<code>\rigidbalanceX</code>	1
<code>\rtab</code>	1
<code>\rtabtext</code>	1

## S

Sacrobosco	12
<code>\sameword</code>	1, 26
<code>\sameword@inedtext</code>	1
Schöpf, Rainer	12
<code>\section@num</code>	1
<code>\select@lemmafont</code>	1
<code>\select@lemmafont</code>	1, 41
<code>\SEref</code>	1, 47
<code>\SErefonlypage</code>	48
<code>\SErefwithpage</code>	1, 48
<code>\series</code>	1
<code>\seriesatbegin</code>	1, 29
<code>\seriesatend</code>	1, 29
<code>\set@line</code>	1
<code>\set@line@action</code>	1
<code>\setapprefprefixmore</code>	48
<code>\setapprefprefixsingle</code>	48
<code>\setcommand@series</code>	1
<code>\sethangingsymbol</code>	1, 43
<code>\setistwofollowinglines</code>	1
<code>\setl@dlp@rbox</code>	1

<code>\setl@drpr@box</code>	1
<code>\setline</code>	1, 20
<code>\setlinenum</code>	1, 21
<code>\setmcellcenter</code>	1
<code>\setmcellleft</code>	1
<code>\setmcellright</code>	1
<code>\setmrowcenter</code>	1
<code>\setmrowleft</code>	1
<code>\setmrowright</code>	1
<code>\setnoteswidthliketwocolumnsX@</code>	1
<code>\setnotesXpositionliketwocolumns@</code>	1
<code>\setprintendlines</code>	1
<code>\setprintlines</code>	1
<code>\setSErefonlypageprefixmore</code>	48
<code>\setSErefonlypageprefixsingle</code>	48
<code>\setSErefprefixmore</code>	48
<code>\setSErefprefixsingle</code>	48
<code>\setsidenotesep</code>	51
<code>\setstanzaindents</code>	1, 41
<code>\setstanzapenalties</code>	1, 43
<code>\setstanzavalues</code>	1
<code>\settcellcenter</code>	1
<code>\settcellleft</code>	1
<code>\settcellright</code>	1
<code>\settoggle@series</code>	1
<code>\setthrowcenter</code>	1
<code>\setthrowleft</code>	1
<code>\setthrowright</code>	1
<code>\setXnotespositionliketwocolumns@</code>	1
<code>\setXnoteswidthliketwocolumns@</code>	1
<code>\showlemma</code>	1, 59
<code>\showwordrank</code>	1, 28
<code>\sidenote@margin</code>	1
<code>\sidenotemargin</code>	1, 50
<code>\sidepstartnumtrue</code>	17
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 21
<code>\splitoff</code>	1
<code>\spreadmath</code>	1, 55
<code>\spreadtext</code>	1, 55
<code>\stanza</code>	1, 41
<code>\stanza@count</code>	1
<code>\stanza@hang</code>	1
<code>\stanza@line</code>	1
<code>\stanzaindent</code>	1, 43
<code>\stanzaindent*</code>	1, 43
<code>\stanzaindentbase</code>	1, 41
<code>\stanzanumwrapper</code>	1, 44
<code>\startlock</code>	1, 20
<code>\startsub</code>	1, 20



<code>\stepl@dcolcount</code> .....	1
<code>\strip@szacnt</code> .....	1
<code>\sub@action</code> .....	1
<code>\sub@lock</code> .....	1
<code>\sub@off</code> .....	1
<code>\sub@on</code> .....	1
<code>\subline@num</code> .....	1
<code>\sublinenumberstyle</code> .....	1, 21
<code>\sublinenumincrement</code> .....	1, 19
<code>\sublinenumr@p</code> .....	1
<code>\sublinenumrep</code> .....	1
<code>\sublineref</code> .....	1, 46
<code>\sublines@false</code> .....	1
<code>\sublines@true</code> .....	1
<code>\sublock@disp</code> .....	1
<code>\sublockdisp</code> .....	1
Sullivan, Wayne .....	12, 13, 60, 73, 77, 154, 155, 231, 265
<code>\sza@penalty</code> .....	1

## T

<code>\tabHilfbox</code> .....	1
<code>\tabhilfbox</code> .....	1
<code>\theadcolcount</code> .....	1
<code>\theadtext</code> .....	1
<code>\theendpageline</code> .....	1
<code>\tfootnoteA</code> .....	28
Theodosius .....	12
<code>\thepageline</code> .....	1
<code>\thepstart</code> .....	1, 17
<code>\thestanza</code> .....	1, 44
<code>\thestartpageline</code> .....	1
<code>\this@line@list@version</code> .....	1
<code>\threecolfootfmt</code> .....	1
<code>\threecolfootfmtX</code> .....	1
<code>\threecolfootgroup</code> .....	1
<code>\threecolfootgroupX</code> .....	1
<code>\threecolfootsetup</code> .....	1
<code>\threecolfootsetupX</code> .....	1
<code>\threecolvfootnote</code> .....	1
<code>\threecolvfootnoteX</code> .....	1
<code>\twocolfootfmt</code> .....	1
<code>\twocolfootfmtX</code> .....	1
<code>\twocolfootgroup</code> .....	1
<code>\twocolfootgroupX</code> .....	1
<code>\twocolfootsetup</code> .....	1
<code>\twocolfootsetupX</code> .....	1
<code>\twocolvfootnote</code> .....	1
<code>\twocolvfootnoteX</code> .....	1

## U

`\unvxhX` ..... 1

## V

Vamana ..... 12  
`\variab` ..... 1  
`\vbfnoteX` ..... 1  
`\vl@dbfnote` ..... 1  
`\vl@dcsnote` ..... 1  
`\vl@dlsnote` ..... 1  
`\vl@drsnote` ..... 1  
`\vnumfootnoteX` ..... 1

## W

Whitney, Ron ..... 12  
`\widthX` ..... 40  
`\wrap@edcrossref` ..... 1  
`\wrapped@bodyfootmarkX` ..... 1  
`\wrapped@footfootmarkX` ..... 1  
Wujastyk, Dominik ..... 11

## X

`\X@doreinfeet` ..... 1  
`\Xafterlemmaseparator` ..... 35  
`\Xafternote` ..... 38  
`\Xafternumber` ..... 33  
`\Xafterrule` ..... 39  
`\Xaftersymmlinenum` ..... 33  
`\Xarrangement` ..... 1, 30  
`\Xarrangement@normal` ..... 1  
`\Xarrangement@paragraph` ..... 1  
`\Xarrangement@threecol` ..... 1  
`\Xarrangement@twocol` ..... 1  
`\Xbeforelemmaseparator` ..... 35  
`\Xbeforenotes` ..... 39  
`\Xbeforenumber` ..... 31, 33  
`\Xbeforesymmlinenum` ..... 33  
`\Xbhookgroup` ..... 39  
`\Xbhooknote` ..... 37  
`\Xboxlinenum` ..... 34  
`\Xboxlinenumalign` ..... 34  
`\Xboxsymmlinenum` ..... 34  
`\Xcolalign` ..... 37  
`\Xdo@feet` ..... 1  
`\xedindex` ..... 1  
`\xedlabel` ..... 1  
`\xedtext` ..... 1  
`\Xendaftererenumber` ..... 33  
`\Xendafterlemmaseparator` ..... 36  
`\Xendafternote` ..... 40

<code>\Xendafterpagenumber</code> .....	35
<code>\Xendaftersymmlinenumber</code> .....	33
<code>\Xendahookinplaceofnumber</code> .....	35
<code>\Xendahooklinenumber</code> .....	35
<code>\Xendbeforelemmaseparator</code> .....	36
<code>\Xendbeforenumber</code> .....	33
<code>\Xendbeforepagenumber</code> .....	35
<code>\Xendbeforesymmlinenumber</code> .....	33
<code>\Xendbhookinplaceofnumber</code> .....	35
<code>\Xendbhooklinenumber</code> .....	35
<code>\Xendbhooknote</code> .....	37
<code>\Xendboxendlinenumalign</code> .....	34
<code>\Xendboxlinenum</code> .....	34
<code>\Xendboxlinenumalign</code> .....	34
<code>\Xendboxstartlinenumalign</code> .....	34
<code>\Xendboxsymmlinenumber</code> .....	34
<code>\Xendhangindent</code> .....	37
<code>\Xendinplaceofflemmaseparator</code> .....	36
<code>\Xendinplaceofnumber</code> .....	34
<code>\Xendlemmadisablefontselection</code> .....	36
<code>\Xendlemmafnt</code> .....	36
<code>\Xendlemmaseparator</code> .....	36
<code>\Xendlineprefixmore</code> .....	35
<code>\Xendlineprefixsingle</code> .....	35
<code>\Xendlinerangeseparator</code> .....	31
<code>\Xendmorethantwolines</code> .....	32
<code>\Xendnonumber</code> .....	32
<code>\Xendnotefontsize</code> .....	37
<code>\Xendnotenumfont</code> .....	36
<code>\Xendnumberonlyfirstinline</code> .....	31
<code>\Xendnumberonlyfirstintwolines</code> .....	31
<code>\Xendparagraph</code> .....	40
<code>\Xendsep</code> .....	40
<code>\Xendsublinesep</code> .....	33
<code>\Xendsymmlinenumber</code> .....	31
<code>\Xendtwolines</code> .....	32
<code>\Xendtwolinesbutnotmore</code> .....	32
<code>\xflagref</code> .....	<u>1</u>
<code>\Xhangindent</code> .....	37
<code>\Xhsizethreecol</code> .....	38
<code>\Xhsizetwocol</code> .....	38
<code>\Xinplaceofflemmaseparator</code> .....	35
<code>\Xinplaceofnumber</code> .....	34
<code>\Xinsertparafootsep</code> .....	<u>1</u>
<code>\Xledsetnormalparstuff</code> .....	<u>1</u>
<code>\xleft@appenditem</code> .....	<u>1</u>
<code>\Xlemmadisablefontselection</code> .....	36
<code>\Xlemmafnt</code> .....	36
<code>\Xlemmaseparator</code> .....	35
<code>\Xlinerangeseparator</code> .....	31

<code>\xlineref</code>	1, 46
<code>\Xmaxhnotes</code>	39
<code>\Xmorethantwolines</code>	31
<code>\Xnolemmaseparator</code>	1, 35
<code>\Xnonbreakableafternumber</code>	33
<code>\Xnonumber</code>	32
<code>\Xnotefontsize</code>	37
<code>\Xnotenumfont</code>	36
<code>\Xnoteswidthliketwocolumns</code>	40
<code>\Xnumberonlyfirstinline</code>	31
<code>\Xnumberonlyfirstintwolines</code>	31
<code>\Xonlypstart</code>	33
<code>\xpageref</code>	1, 46
<code>\Xparafootsep</code>	38
<code>\Xparindent</code>	37
<code>\Xpstart</code>	32
<code>\Xpstarteverytime</code>	32
<code>\xpstartref</code>	1, 46
<code>\XR@test</code>	1
<code>\XR@test@mac</code>	1
<code>\XR@test@mac@test</code>	1
<code>\Xragged</code>	38
<code>\xright@appenditem</code>	1
<code>\Xrigidbalance</code>	1
<code>\Xstanza</code>	33
<code>\Xstanzaseparator</code>	33
<code>\xsublineref</code>	1, 46
<code>\Xsublinesep</code>	33
<code>\Xsymlinenum</code>	31
<code>\Xtwolines</code>	31
<code>\Xtwolinesonlyinsamepage</code>	32
<code>\Xtxtbeforenotes</code>	39
<code>\Xunvxh</code>	1
<code>\Xwidth</code>	40
<code>\xxref</code>	1, 47
<b>Z</b>	
<code>\zz@@@</code>	1

## Change History

v0.1.0.		
General: First public release	.....	1
v0.2.0.		
General: Added tabmac code, and extended indexing	.....	1
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	.....	66
\morenoexpands: Added \l@dtabnoexpands to \no@expands	.....	111
\reledmac@error: Added \eledmac@error and replaced error messages	.....	67
v0.2.1.		
\@lab: Removed page setting from \@lab	.....	234
General: Added text about normal labeling	.....	47
Bug fixes and match with mempatch v1.8	.....	1
Major changes to insert code when memoir is loaded	.....	229
\doxtrafeet: Renamed \doxtrafeet to \l@ddoxtrafeet	.....	226
\edlabel: Tweaked \edlabel to get correct page numbers	.....	232
\l@ddodoreinextrafeet: Renamed \dodoreinextrafeet to \l@ddodoreinextrafeet	.....	227
\morenoexpands: Removed some \lets from \no@expands. These were in edmac but Peter Wilson feels that they should not have been as they disabled page/line refs in a footnotes	.....	111
\zz@@@: Minor change to \zz@@@	.....	231
v0.2.2.		
General: Improved paragraph footnotes	.....	1
New Dekker example	.....	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts pack- age	.....	72
\footfudgefiddle: Added \footfudgefiddle	.....	152
\line@list@stuff: Added initial write of page number in \line@list@stuff	...	104
\para@footsetup: Added \footfudgefiddle to \para@footsetup	.....	152
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	.....	189
v0.3.0.		
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	.....	234
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	.....	93
General: Includes edstanza and more	.....	1
\ledlinenum: Added \linenumr@p and \sublinenumr@p to \leftlinenum and \rightlinenum .....	.....	85
\linenumberlist: Added \linenumberlist mechanism	.....	73
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	..	205
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	.....	174
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	.....	84
v0.3.1.		
General: Not released. Added remarks about the parallel package	.....	1
v0.4.0.		
\@iiiminipage: Modified kernel \@iiiminipage and \endminipage to cater for criti- cal footnotes	.....	253
General: Added final/draft options	.....	64
Added ledgroup environment	.....	254
Added ledgroupsize environment	.....	254
Added minipage, etc., support	.....	1

<code>\edtext</code> : Added <code>\showlemma</code> to <code>\edtext</code> .....	112
<code>\l@dfteetendmini</code> : Added <code>\l@dfteetbeginmini</code> , <code>\l@dfteetendmini</code> and all their supporting code .....	251
<code>\mpnormalfootgroup</code> : Added <code>\mpnormalfootgroup</code> .....	151
<code>\mpnormalvfootnote</code> : Added <code>\mpnormalvfootnote</code> .....	149
<code>\showlemma</code> : Added <code>\showlemma</code> .....	73
<code>\Xarrangement@normal</code> : Added minpage footnote setup to <code>\footnormal</code> .....	148
v0.4.1.	
General: Added code for changing <code>\@docclearpage</code> .....	230
Not released. Minor editorial improvements and code tweaks .....	1
Only change <code>\@footnotetext</code> and <code>\@footnotemark</code> if memoir not used .....	175
<code>\edindex</code> : Let <code>eledmac</code> take advantage of memoir's indexing .....	259
<code>\print@Xnotes</code> : Added <code>\@opXfeet</code> .....	227
<code>\Xdo@feet</code> : Changed <code>\Xdo@feet</code> code for easier extensions .....	227
v0.5.0.	
<code>\@footnotetext</code> : Enabled regular <code>\footnote</code> in numbered text .....	176
<code>\@xympar</code> : Eliminated <code>\marginpar</code> disturbance .....	245
General: Added left and right side notes .....	245
Added sidenotes, familiar footnotes in numbered text .....	1
v0.5.1.	
General: Added moveable side note .....	245
Fixed right line numbers killed in v0.5 .....	1
Only change <code>\hsize</code> in <code>ledgroupsize</code> environment otherwise page number can be in wrong place .....	254
<code>\affixline@num</code> : Changed <code>\affixline@num</code> to cater for sidenotes .....	136
<code>\l@dgetsidenote@margin</code> : Added <code>\sidenotemargin</code> and <code>\sidenote@margin</code> ...	245
v0.6.0.	
<code>\@lopR</code> : Added <code>\@pend</code> , <code>\@pendR</code> , <code>\@lopL</code> and <code>\@lopR</code> in anticipation of parallel processing .....	95
<code>\@nl@reg</code> : Added <code>\fix@page</code> to <code>\@nl</code> .....	93
Extended <code>\@nl</code> to include the page number .....	93
General: Fixed long paragraphs looping .....	1
Fixed minor typos .....	1
Prepared for <code>eledpar</code> package .....	1
<code>\fix@page</code> : Added <code>\last@page@num</code> and <code>\fix@page</code> .....	95
<code>\get@thisfootnote</code> : Changed <code>\l@dbfnote</code> and <code>\v1@dbfnote</code> as originals could give incorrect markers in the footnotes .....	176
<code>\new@line</code> : Extended <code>\new@line</code> to output page numbers .....	105
v0.7.0.	
<code>\@nl@reg</code> : Added <code>\@nl@reg</code> .....	93
<code>\@ref@reg</code> : Added <code>\@ref@reg</code> .....	101
General: <code>eledmac</code> having been available for 2 years, deleted the commented out original <code>edmac</code> texts .....	1
Maïeul Rouquette new maintainer .....	1
Made macros of all messages .....	67
Replaced all <code>\interfootnotelinepenalty</code> , etc., by just <code>\interfootnotelinepenalty</code> .....	1
Tidying up for <code>eledpar</code> and <code>ledarab</code> packages .....	1
<code>\affixline@num</code> : Added skipnumering to <code>\affixline@num</code> .....	136
<code>\do@actions@fixedcode</code> : Added <code>\do@actions@fixedcode</code> .....	134

<code>\do@actions@next</code> : Added number skipping to <code>\do@actions</code> .....	133
<code>\do@insidelinehook</code> : Added <code>\do@linehook</code> for use in <code>\do@line</code> .....	131
<code>\endnumbering</code> : Changed <code>\endnumbering</code> for <code>eledpar</code> .....	76
<code>\fx@l@cks</code> : Added <code>\ch@cksub@l@ck</code> , <code>\ch@ck@l@ck</code> and <code>\fx@l@cks</code> .....	138
<code>\footsplitskips</code> : Added <code>\footsplitskips</code> for use in many footnote styles .....	146
<code>\get@linelistfile</code> : Added <code>\get@linelistfile</code> .....	92
<code>\initnumbering@reg</code> : Added <code>\initnumbering@reg</code> .....	75
<code>\l@advance@parledgroup@beforenormalnotes</code> : Added <code>\l@dunboxmpfoot</code> containing some common code .....	253
<code>\l@dcsn@text@r</code> : Added <code>\l@emptyd@ta</code> .....	131
<code>\l@dgetline@margin</code> : Added <code>\l@dgetline@margin</code> .....	81
<code>\l@dgetlock@disp</code> : Added <code>\l@dgetlock@disp</code> .....	83
<code>\l@dgetsidenote@margin</code> : Added <code>\l@dgetsidenote@margin</code> .....	245
<code>\l@dnumpstartsL</code> : Added <code>\l@dnumpstartsL</code> , <code>\ifl@dpairing</code> and <code>\ifpst@rted</code> for/from <code>eledpar</code> .....	73
<code>\l@drsn@te</code> : Added <code>\l@dlsn@te</code> and <code>\l@drsn@te</code> for use in <code>\do@line</code> .....	132
<code>\l@dzeropenalties</code> : Added <code>\l@dzeropenalties</code> .....	127
<code>\ledlinenum</code> : Added <code>\ledlinenum</code> for use by <code>\leftlinenum</code> and <code>\rightlinenum</code> ..	85
<code>\line@list@stuff</code> : Deleted <code>\page@start</code> from <code>\line@list@stuff</code> .....	104
<code>\list@clearing@reg</code> : Added <code>\list@clearing@reg</code> .....	91
<code>\n@num</code> : Added <code>\n@num</code> .....	100
<code>\normalbfnoteX</code> : Removed extraneous space from <code>\normalbfnoteX</code> .....	182
<code>\resumenumbering</code> : Changed <code>\resumenumbering</code> for <code>eledpar</code> .....	77
<code>\setprintendlines</code> : Added <code>\setprintendlines</code> for use by <code>\printendlines</code> ...	203
<code>\setprintlines</code> : Added <code>\setprintlines</code> for use by <code>\printlines</code> .....	170
<code>\skipnumbering</code> : Added <code>\skipnumbering</code> and supports .....	108
<code>\sublinenumincrement</code> : Added <code>\firstlinenum</code> , <code>\linenumincrement</code> , <code>\firstsublinenum</code> and <code>\linenumincrement</code> .....	82
<code>\sublinenumr@p</code> : Using <code>\linenumrep</code> instead of <code>\linenumr@p</code> .....	84
Using <code>\sublinenumrep</code> instead of <code>\sublinenumr@p</code> .....	84
<code>\vnumfootnoteX</code> : Removed extraneous space from <code>\vnumfootnoteX</code> .....	184
v0.8.0.	
General: Bug on endnotes fixed: in a // text, all endnotes will print and be placed at the ends of columns () .....	1
v0.8.1.	
General: Bug on <code>\edtext</code> ; <code>\critex</code> ; <code>\lemma</code> fixed: we can now us non-switching commands .....	1
v0.9.0.	
General: No more <code>ledpatch</code> . All patches are now in the main file. ....	1
v0.9.1.	
General: Fix some bugs linked to integrating <code>ledpatch</code> on the main file. ....	1
v0.10.0.	
General: Corrections to <code>\section</code> and other titles in numbered sections .....	1
v0.11.0.	
General: Makes it possible to add a symbol on each verse's hanging, as in French typography. Redefines the command <code>\hangingsymbol</code> to define the character. ....	1
v0.12.0.	
General: For compatibility with <code>eledpar</code> , possibility to use <code>\autopar</code> on the right side. ...	1
Possibility to number <code>\pstart</code> . ....	17
Possibility to number the <code>pstart</code> with the commands <code>\numberpstarttrue</code> . ....	1

\l@dnumpstartsL: Added \ifledRcol and \ifnumberingR for/from eledpar . . . .	73
v0.12.1.	
General: Don't number \pstarts of stanza. . . . .	1
The numbering of \pstarts restarts on each \beginnumbering. . . . .	1
v0.13.0.	
General: New stanzaindentsrepetition counter to repeat stanza indents every $n$ verses. . . .	42
New stanzaindentsrepetition counter: to repeat stanza indents every $n$ verses. . . . .	1
\managestanza@modulo: New stanzaindentsrepetition counter to repeat stanza indents every $n$ verses. . . . .	267
v0.13.1.	
General: \thepstartL and \thepstartR use now \bfseries and not \bf, which is deprecated and makes conflicts with memoir class. . . . .	1
v0.14.0.	
General: Tweaked \edlabel to get correct line number if the command is first element of a paragraph. . . . .	1
\edlabel: Tweaked \edlabel to get correct line number if the command is first element of a paragraph. . . . .	232
v0.15.0.	
General: Line numbering can be reset at each pstart. . . . .	79
Possibility to print \pstart number inside. . . . .	17
\affixline@num: Line numbering can be disabled. . . . .	136
\ifinserthangingsymbol: New management of hangingsymbol insertion, preventing undesirable insertions. . . . .	266
\printlines: Line numbering can be reset at each pstart. . . . .	173
v0.17.0.	
\ifinserthangingsymbol: New new management of hangingsymbol insertion, preventing undesirable insertions. . . . .	266
v1.0.0.	
General: \lemma can contain commands. . . . .	24
Debug in lineation command . . . . .	19
New generic commands to customize footnote display. . . . .	30, 218
Options nonum and nosep in \Xfootnote. . . . .	23
Options of \Xfootnotes. . . . .	144
Possibility to have commands in sidenotes. . . . .	50
Some compatibility break with eledmac. Change of name: eledmac. . . . .	1
\morenoexpands: Change to be compatible with new features . . . . .	111
v1.0.1.	
General: Correction on \Xnumberonlyfirstinline with lineation by pstart or by page. 31	
v1.1.0.	
General: Add \labelpstarttrue. . . . .	17
Add \Xnumberonlyfirstintwolines . . . . .	31
Add \Xpstart and \Xonlypstart . . . . .	32
New hook to add arbitrary code at the beginning of the notes . . . . .	37
New options for block of notes. . . . .	39
New package option: parapparatus. . . . .	1
New tools to change order of series . . . . .	217
Sectioning commands. . . . .	57
\preXnotes: New skip \preXnotes@ . . . . .	197
\settoggle@series: \settoggle@series switch the global value of the toggle, not only the local value. . . . .	218



v1.2.0.	
<code>\endquote</code> : Compatibility of <code>\ledchapter</code> with the <i>memoir</i> class. . . . .	294
<code>\preXnotes</code> : Debug in familiar footnotes (bug introduced by v1.1). . . . .	197
v1.3.0.	
<code>\endquote</code> : <i>Quotation</i> and quote environment inside numbered sections. . . . .	294
v1.4.0.	
General: Compatibility with LuaTeX of RTL notes. . . . .	1
<code>\edtext</code> : Compatibility of <code>\edtext</code> with the right-to-left direction (with Polyglossia). . . . .	112
<code>\ledsetnormalparstuffX</code> : Direction of footnotes with polyglossia. . . . .	194
<code>\newseries@</code> : Remembers the language of the lemma, in order to create a correct direction for the footnote separator. . . . .	209
<code>\rbracket</code> : Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). . . . .	165
v1.4.1.	
<code>\affixside@note</code> : Remove spurious spaces. . . . .	250
<code>\endquote</code> : New option <i>noquotation</i> . . . . .	294
<code>\get@thisfootnote</code> : Compatibility of standard footnotes with <code>eledmac</code> when these footnotes contain any commands. . . . .	176
<code>\labelrefsparsesubline</code> : Fix bug with <code>\edlabel</code> . . . . .	233
v1.4.2.	
General: Debug with some special classes. . . . .	1
v1.4.3.	
General: Add <code>\Xnonbreakableafternumber</code> . . . . .	33
Spurious space after familiar footnotes. . . . .	1
v1.4.4.	
General: Label inside familiar footnotes. . . . .	1
v1.4.5.	
General: Bug with <code>komasscript</code> + <code>eledpar</code> + <code>chapter</code> . . . . .	1
v1.4.6.	
General: Bug with <i>memoir</i> class introduced by 1.4.5. . . . .	1
v1.4.7.	
<code>\endquote</code> : Compatibility of sectioning commands with <code>\autopar</code> . . . . .	294
v1.4.8.	
General: Corrects a bug with parallel texts introduced by 1.1. . . . .	1
v1.4.9.	
<code>\normalbfnoteX</code> : Allow to redefine <code>\thefootnoteX</code> with <code>alph</code> when some packages are loaded. . . . .	182
v1.5.0.	
General: Correct indexing when the call is made in critical notes. . . . .	256
<code>\do@insidelinehook</code> : Added <code>\do@insidelinehook</code> for use in <code>\do@line</code> . . . . .	131
<code>\edindex</code> : Compatibility with <code>imakeidx</code> package, and possibility to use multiple index with <code>\edindex</code> . . . . .	259
v1.5.1.	
<code>\managestanza@modulo</code> : Correct <code>stanzaindentsrepetition</code> counter . . . . .	267
<code>\normalvfootnoteX</code> : Fix bug with normal familiar footnotes when mixing RTL and LTR text. . . . .	179
v1.6.0.	
<code>\newverse</code> : Add <code>\falseverse</code> macro. . . . .	270
v1.6.1.	
General: Corrects a false hanging verse when a verse is exactly the length of a line. . . . .	1

<code>\AtEveryPstart</code> : Spurious space in <code>\pstart</code> . . . . .	124
<code>\ifinserthangingsymbol</code> : Hang verse is now not automatically flush right. . . . .	266
<code>\l@dunhbox@line</code> : Move the call to <code>\inserthangingsymbol</code> to allow use <code>\hfill</code> inside. . . . .	129
<code>\pend</code> : Spurious space in <code>\pend</code> . . . . .	125
v1.7.0.	
General: New features for managing page breaks. . . . .	58
v1.8.0.	
General: Compatibility with <code>parledgroup</code> option of <code>eledpar</code> package. . . . .	1
If <code>imakeidx</code> and <code>hyperref</code> are loaded, adds <code>hyperref</code> in the index. . . . .	256
<code>\endquote</code> : Correction of sectioning commands in parallel texts. . . . .	294
<code>\get@index@command</code> : Debug <code>\get@index@command</code> and compatibility with <code>hyperref</code> package. . . . .	258
<code>\newhookcommand@series@reload</code> : Debug <code>\beforenotesX</code> and <code>\maxhnotesX</code> which did not work. . . . .	221
<code>\prevpage@num</code> : Correct <code>\parafootsep</code> when using with <code>ledgroup</code> . . . . .	158
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined). . . . .	199
v1.8.2.	
General: Debug compatibility problem with <code>hebrew</code> option of <code>babel</code> package. . . . .	1
v1.8.3.	
General: Fixes spurious spaces added by v1.7.0. . . . .	1
v1.8.5.	
General: Debug indexing in right column, with <code>eledpar</code> . . . . .	256
v1.9.0.	
<code>\doxtrafeet</code> : Add <code>\fnpos</code> to choice the order of footnotes. . . . .	226
<code>\l@dfeetendmini</code> : Add <code>\mpfnpos</code> to choice the order of footnotes in <code>minipage</code> / <code>ledgroup</code> . . . . .	251
v1.10.0.	
General: Add <code>\pstartref</code> and <code>\xpstartref</code> to refer to a <code>pstart</code> number (extension of <code>\edlabel</code> ). . . . .	1
<code>\endquote</code> : Correction of sectioning commands in parallel texts. . . . .	294
v1.10.1.	
General: Compatibility with <code>cleveref</code> . . . . .	1
v1.10.2.	
General: Compatibility of <code>stanza</code> with v1.8a of <code>babel-greek</code> . . . . .	1
v1.10.3.	
General: Debug of cross-referencing. . . . .	1
v1.10.4.	
General: Debug of critical notes in <code>edtabular</code> environment. . . . .	1
v1.10.5.	
General: Debug of <code>\pausenumbering</code> . . . . .	1
Debug of <code>\xxref</code> . . . . .	1
v1.10.6.	
General: Debug of interaction between <code>\autopar</code> and <code>\pausenumbering</code> . . . . .	1
v1.11.0.	
General: Add hooks to disable the font selection for lemma in footnote. . . . .	36
v1.11.1.	
General: Correct a bug when a critical note starts with plus or minus. . . . .	1

v1.12.0.	
\@nl@reg: To ensure compatibility with <code>\musixtex</code> , <code>\@1</code> becomes <code>\@1</code> . Consequently, \@1@reg becomes \@nl@reg. . . . .	93
General: Add <code>\ledinnernote</code> and <code>\ledouternote</code> commands. . . . .	50
Add <code>\Xendparagraph</code> and related settings. . . . .	40
Add hyperlink to <code>crossref</code> (needs <code>hyperref</code> package). . . . .	45
Compatibility with <code>musixtex</code> . . . . .	1
Debug <code>eledmac</code> sectioning command after using <code>\resumenumbering</code> . . . . .	1
Ensure that <code>imakeidx</code> is loaded <i>before</i> <code>eledmac</code> . . . . .	256
New hooks: <code>\Xafterrule</code> and <code>\afterruleX</code> . . . . .	39
New options for ragged-paragraph notes . . . . .	38
New sectioning commands. . . . .	57
Optional arguments for <code>\pstart</code> and <code>\pend</code> . . . . .	17
\AtEveryPstart: New optional argument for <code>\pstart</code> , to execute code before it. . . . .	124
\edindex: Use correctly default index when <code>imakeidx</code> is loaded. . . . .	259
\endquote: <code>\ledxxx</code> sectioning commands are deprecated and replaced by <code>\eledxxx</code> commands. . . . .	294
\initnumbering@reg: <code>\beginnumbering</code> is defined only on <code>eledmac</code> , not on <code>eledpar</code> . . . . .	75
\l@dcsnote: <code>\l@dlsnote</code> , <code>\l@drsnote</code> and <code>\l@dcsnote</code> defined only one time, in <code>eledmac</code> , including needs for <code>eledpar</code> case. . . . .	247
\l@dgetsidenote@margin: <code>\sidenotemargin</code> is now directly defined in <code>eledmac</code> to be able to manage <code>eledpar</code> . . . . .	245
\l@dnumpstartsL: Add <code>\ifledRcol@</code> for <code>eledpar</code> . . . . .	73
\l@dunhbox@line: <code>\do@line</code> is split in more little commands. . . . .	129
\newhookcommand@series@reload: Debug <code>\beforenotesX</code> and <code>\maxhnotesX</code> which did not work when called after <code>\footparagraphX</code> . . . . .	221
Debug <code>\Xbeforenotes</code> and <code>\Xmaxhnotes</code> which did not work when called after <code>\footparagraph</code> . . . . .	221
\pend: New optional argument for <code>\pend</code> , to execute code after it. . . . .	125
\stanza: <code>&amp;can</code> have an optional argument: content to be printed after. . . . .	270
<code>\Stanza</code> can have an optional argument: content to be printed before. . . . .	270
Add <code>\newverse</code> macro, <code>\falseverse</code> deprecated. . . . .	270
v1.12.1.	
\wrap@edcrossref: Fix spurious spaces. . . . .	236
v1.12.2.	
\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0)	129
v1.12.3.	
General: Add macros for new messages since v0.7 . . . . .	67
Correct bug with side and familiar notes in tabular environments. . . . .	1
Debug <code>\eledxxx</code> with some paper size . . . . .	1
Debug <code>\ledinnernote</code> and <code>\ledouternote</code> commands in the top of pages. . . . .	50
Debug left and right notes (bugs added by 1.12.0) . . . . .	1
Underline lemma in <code>\eledxxx</code> when using draft mode. . . . .	1
\flag@end: <code>\flag@start</code> and <code>\flag@end</code> are now defined only one time for <code>eledmac</code> and <code>eledpar</code> . . . . .	106
<code>\flag@start</code> send a error message when a <code>\edtext</code> is done without insert (note)	106
\reledmac@error: Replaced error messages . . . . .	67
v1.12.4.	
General: Debug spurious page breaks before <code>\chapter</code> (bug added in 1.12.0) . . . . .	1

v1.12.5.	
\@edindex@hyperref: Debug \edindex when hyperref is not loaded	261
\@ssect: Debug \eledchapter in parallel with memoir	297
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	131
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	76
\l@gobblearg: \l@gobblearg becomes \l@gobbelloptarg	277
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	278
\resumenumbering: Debug \resumenumbering	77
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	236
v1.12.8.	
\flag@end: \flag@start do not send a error message when a \edtext is done without insert (note) but have a endnote	106
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	40
Added widthliketwocolumns option	64
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	221
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	189
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	218
v1.13.1.	
General: Coming back of page and line breaking penalties's management, deleted by error in v0.17.	1
Debug quotation environment inside of a \pstart preceded by a sectioning command.	1
\thepstart: Add \l@dzeropenalties in \pstart	124
v1.13.2.	
General: Fix bug with normal footnotes, added by v1.13.0.	1
\l@dnumpstartsL: Add \ifl@dpaging for eledpar	73
v1.13.3.	
General: Fix extra spaces with paragraphed footnotes, added by v1.13.0.	1
v1.13.4.	
General: Fix bug with index when memoir class is used without hyperref	1
v1.14.0.	
General: Debug spurious characters before endnotes.	199
Delete previous override of \l@d@@wrindexhyp at the beginning of a document when hyperref is not loaded.	263
Move gobbling command	72
Provide \@gobblefour	72
\edindex: Let eledmac take advantage of imakeidx even when memoir class is used	259
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref package.	300
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	297
v1.15.0.	
\@edtext@level: New boolean \if@edtext@.	111
General: Fix bug with footnotes layout when using some options of the geometry package (bug add by v1.13.0).	1
New commands \AtEveryPstart and \AtEveryPend.	17

New tools to prevent ambiguous references in lemma	26
<code>\arrangementX@threecol</code> : Correct bug with paragraphed familiar footnotes setting.	189
<code>\endsub</code> : Restore subline feature (disabled by mistake in v1.8.0).	106
<code>\if@lemmacommand@</code> : New boolean <code>\iflemmacommand@</code> .	117
v1.15.1.	
<code>\line@list@stuff</code> : Revert modification of 1.5.2 which makes bug with numbering.	
Leave vertical mode to solve spurious space before minipage.	104
v1.16.0.	
General: <code>\edtext</code> is now defined only in <code>eledmac</code> , not in <code>eledpar</code> . Debug wrong numbering when using <code>\sameword + eledpar + \tag</code> command.	112
Compatibility of standard footnotes with some biblatex styles.	1
New <code>\stanzaindent</code> command.	1
v1.16.1.	
<code>\xlineref</code> : <code>\lineref</code> is not defined if defined by some other package, like <code>lineno</code> .	
Eledmac provides <code>\edlineref</code> instead.	236
v1.17.0.	
<code>\edtext</code> : Error message when calling <code>\edtext</code> outside of a numbered paragraph.	112
v1.18.0.	
<code>@edindex@hyperref</code> : Fix spurious space with <code>\edindex</code> when using <code>imakeidx/indextools + hyperref</code> .	261
General: Add <code>\Xpstarteverytime</code>	32
Compatibility with Lua $\TeX$ RTL languages.	1
Debug <code>\Xonlypstart</code> when using <code>\Xnumberonlyfirstinline</code> and the current line number differs from the previous.	32
<code>\edlabel</code> : <code>\edlabel</code> is now defined only one time for both <code>eledmac</code> and <code>eledpar</code>	232
<code>\l@d@section</code> : Option <code>parapparatus</code> works for endnotes.	200
<code>\l@dnumpstartsL</code> : Add <code>\ifl@dprintingpages</code> and <code>\dprintingcolumns</code> for <code>eledpar</code>	73
<code>\print@line</code> : Compatibility with Lua $\TeX$ RTL languages.	129
<code>\printlinefootnote</code> : Code refactoring in <code>\printlinefootnote</code> : the printing of the numbers are factorized in <code>\printlinefootnotearea</code>	166
<code>\printpstart</code> : Debug <code>\Xpstart</code> with parallel pages and columns ( <code>eledpar</code> )	166
v1.19.0.	
General: <code>\Xmaxhnotes</code> and <code>\maxhnotesX</code> work now for both two-columns and three-columns setting.	1
Compatibility with <code>eledpar v1.13.0</code> .	1
<code>\footsplitskips</code> : <code>\footsplitskips</code> doesn't set <code>\floatingpenalty</code> to <code>\@MM</code> when processing parallel pages.	146
<code>\xxref</code> : <code>\xxref</code> works also with right side numbers, when <code>\@Rlineflag</code> is not empty.	238
v1.19.1.	
General: Call <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> directly in <code>\print@notesX@forpages</code> and <code>\print@Xnotes@forpages</code> , that is in <code>eledpar</code> .	1
v1.20.0.	
General: Add <code>\Xendboxlinenum</code>	34
Add <code>\Xtwolines</code> and <code>\Xmorethantwolines</code> hooks	31
Add series option.	1
Correct <code>\Xinplaceofnumber</code> hook.	1
Explicit error message when calling <code>\Xfootnote</code> outside of <code>\edtext</code> .	1
Fix bug with line number typesetting direction when using <code>\eledsection</code> and similar commands for RTL texts with Lua $\TeX$ .	1

Fix issues with RTL text in notes when using Lua $\TeX$ .	1
Options fulllines in $\backslash$ Xfootnote.	23
The $\backslash$ newifs are not followed by boolean values set to false, because it is the $\TeX$ default setting.	1
$\backslash$ printlines: Added $\backslash$ ifl@d@Xmorethantwolines and $\backslash$ ifl@d@Xmorethantwolines to $\backslash$ printlines	174
$\backslash$ stanza: & and $\backslash$ & can be preceded by spaces.	270
$\backslash$ xxref: Debug $\backslash$ xxref when not loading eledpar (fix bug added in 1.19.0).	238
v1.21.0.	
$\backslash$ edindex@hyperref: Look at the hyperindex option of hyperref before inserting hyperref	261
General: $\backslash$ AtEveryPstart and $\backslash$ AtEveryPend are now compatible with $\backslash$ autopar	1
$\backslash$ Xafterrule and $\backslash$ afterruleX features no longer create problems of overflowing at the bottom of the page.	1
$\backslash$ chapter inside optional argument of $\backslash$ pstart works when typesetting parallel pages	1
$\backslash$ preXnotes and $\backslash$ prenotesX features no longer create problems of overflowing at the bottom of the page.	1
$\backslash$ seriesatbegin and $\backslash$ seriesatbegin more efficient	217
Add $\backslash$ applabel and related	48
Add $\backslash$ beforenotesX and $\backslash$ Xbeforenotes features for notes set in two and three column.	1
Add $\backslash$ hidenumbers	21
Add $\backslash$ Xcolalign and $\backslash$ colalignX hooks	37
Add $\backslash$ Xendtwolines, $\backslash$ Xendmorethantwolines, $\backslash$ Xendtwolinesbutnotmore and $\backslash$ Xendtwolinesonlyinsamepage.	32
Add $\backslash$ Xparindent and $\backslash$ hangindentX	37
Add $\backslash$ Xtwolinesbutnotmore and $\backslash$ Xtwolinesonlyinsamepage.	1
Add nocritical, noend, nofamiliar and noledgroup options.	1
Add noeledsec package option	1
Debug $\backslash$ beforenotesX $\backslash$ maxhnotesX $\backslash$ noteswidthliketwocolumnsX and $\backslash$ afterruleX with footnotes set in two and three columns.	1
Fix bug when a $\backslash$ Xfootnote follows a $\backslash$ Xendnote in the second argument of $\backslash$ edtext (bug added in eledmac 1.0.0).	1
Fix bug with $\backslash$ maxhnotesX when using $\backslash$ foottwocolX or $\backslash$ footthreecolX.	1
Fix bug with space between columns with notes in two columns (bug added in v1.13.0).	1
Fix spurious space after first page number in $\backslash$ doendnotes. oldprintnpspace option allows to come back to previous setting	1
parapparatus option works now with familiar footnotes.	1
Provide $\backslash$ @gobblefive	72
$\backslash$ l@d@section: $\backslash$ endnotes take five arguments.	200
$\backslash$ ledinnotemark: Add $\backslash$ ledinnotemark.	259
$\backslash$ ledsetnormalparstuffX: $\backslash$ ledsetnormalparstuff is deprecated and becomes $\backslash$ ledsetnormalparstuffX and $\backslash$ Xledsetnormalparstuff.	194
$\backslash$ n@num: $\backslash$ n@num@ref deleted	100
$\backslash$ n@num defined only one time for both Eledmac and Eledpar	100
$\backslash$ newhookcommand@series: $\backslash$ newhookcommand@series can take an optional argument.	220
$\backslash$ newhooktoggle@series: $\backslash$ newhooktoggle@series can take an optional argument.	220

<code>\print@footnoteXrule</code> : Code refactoring: the spaces after the footnote rules are directly managed in <code>\print@Xfootnoterule</code> and <code>\print@footnoteXrule</code> . . . . .	196
<code>\seriesatend</code> : Fix spurious space in <code>\seriesatend</code> . . . . .	217
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code> . . . . . .	108
Correct <code>\skipnumbering</code> for stanza. . . . .	108
Delete <code>\skipnumbering@reg</code> . . . . .	108
v1.22.0.	
General: Add <code>\doendnotesbysection</code> command. . . . .	24
Add option for lemma separator inside endnotes . . . . .	36
Adds hyperlink for references to notes in indices. . . . .	1
Fix conflict between <code>noend</code> package option and <code>edtabularx</code> environments . . . . .	1
Provides support for <code>xindy</code> . . . . .	1
Standardize endnotes handbook. . . . .	24
When using <code>hyperref</code> package, internal links in index or with <code>\edlineref</code> are now targeted to the top and not longer to the bottom of the lines they refer to. . . . .	1
<code>\ledinnote</code> : <code>\ledinnote</code> takes a first optional argument, which is the label for hyperlinks. . . . .	259
v1.22.1.	
General: Fix bug (added on v1.22.0) with <code>\Xinplaceofnumber</code> hook. . . . .	1
<code>\prevpage@num</code> : Correct double symbol when using both <code>\parafootsep</code> and <code>\Xsymlinenum</code> . . . . . .	158
v1.23.0.	
<code>\@edtext@level</code> : The boolean <code>\if@edtext@</code> becomes the counter <code>\edtext@level</code> . . . . .	111
General: Add <code>\Xboxlinenumalign</code> and <code>\Xendboxlinenumalign</code> . . . . .	34
Add <code>\Xboxstartlinenum</code> , <code>\Xendboxstartlinenum</code> , <code>\Xboxendlinenum</code> , <code>\Xendboxendlinenum</code> . . . . . .	34
Allow use of <code>\sameword</code> with <code>inputenc</code> managing of UTF-8. . . . .	1
Compatibility between <code>nofamiliar/nocriticals</code> option and <code>minipage/ledgroup</code> . . . . .	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code> . . . . .	1
Fix bug with <code>\sameword</code> when the lemma overlaps multiple line. . . . .	26
Fix bug with <code>\sameword</code> when the same lemma is used for multiple notes or for nested <code>\edtexts</code> . . . . .	26
Fix bug with <code>\skipnumbering</code> called immediately after a <code>\pstart</code> . . . . .	1
Fix error of <code>\iftrue</code> not closed. . . . .	1
Fix spurious space with <code>\skipnumbering</code> (bug added on v1.21.0). . . . .	1
New tools to ensure the line-list file uses the right version of commands when upgrading the <code>eledmac</code> version. . . . .	1
Optional argument of <code>\sameword</code> can be a comma-separated list of <code>\edtext</code> depth. . . . .	26
<code>\lemma</code> : Fix spurious space after <code>\lemma</code> command . . . . .	116
<code>\newseries@</code> : Prevent spurious spaces when <code>\Afootnote</code> and similar commands are followed by spaces (bug added on 1.0.0). . . . .	209
<code>\sameword</code> : In order to allow use of <code>\sameword</code> with <code>inputenc</code> , we detokenize its mandatory argument before using it in control sequence names. . . . .	120
<code>\Serefwithpage</code> : Debug <code>\Xendtwolines</code> , <code>\Xendmoreethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> when using <code>\apprefwithpage</code> . . . . .	241
v1.23.1.	
General: Fix bug with <code>\lemma</code> command in the right side. . . . .	1
v1.23.2.	
General: Compatibility with L <sup>A</sup> T <sub>E</sub> X's release 2015. . . . .	1

v1.24.0.	
General: We can reinitialize <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> providing to it an empty argument. . . . .	1
v1.24.1.	
General: <code>\lemma</code> is disabled when using ‘nocritical’ option. . . . .	1
v1.24.2.	
General: Fix incompatibility between ‘nofamiliar’ option and ‘memoir’ package. . . . .	1
v1.24.3.	
General: Restore marginal numbers and notes with sectioning command (bug introduced in v1.21.0) . . . . .	1
v1.24.4.	
General: Fix spurious space with <code>\edindex</code> when using <code>xindy+hyperref</code> option. . . . .	1
v1.24.5.	
General: Fix bug of indent, when a added in 1.1.0, when a <code>\beginnumbering</code> immediately follow a sectioning command. . . . .	1
v2.0.0.	
<code>\@iiiminipage</code> : Patch <code>\@iiiminipage</code> instead of redefining it. . . . .	253
<code>\@xympar</code> : Patching <code>\@xympar</code> instead of redefining it . . . . .	245
General: <code>\@makecol</code> , <code>\@reinserts</code> and <code>\@doclearpage</code> are patched instead of begin redefined . . . . .	229
<code>\doxtrafeeti</code> becomes <code>\do@feetX</code> ; <code>\doxtrafeetii</code> becomes <code>\Xdo@feet</code> ; <code>\@opxtrafeeti</code> becomes <code>\@opfeetX</code> ; <code>\doreinxtrafeetii</code> becomes <code>\X@doreinfeet</code> ; <code>\doreinxtrafeeti</code> becomes <code>\@doreinfeetX</code> . . . . .	229
Add <code>\Xendinplaceofnumber</code> hook. . . . .	1
Add <code>\Xendnonumber</code> hook. . . . .	1
Add nonum option for endnotes. . . . .	1
Fix bug when printing only one series of endnotes, but wanted to keep endnotes for other series. . . . .	1
In order to have a more consistent name’s convention, many names has been changed. . . . .	1
Many $\TeX$ ’s output macros are now patched and not override. . . . .	1
Package’s name becomes <code>reledmac</code> . . . . .	1
Patch <code>\@footnotemark</code> instead of redefine it . . . . .	176
Suppress indexing command specific to <code>memoir</code> . . . . .	259
<code>\endminipage</code> : Patch <code>\endminipage</code> instead of redefining it. . . . .	253
<code>\initnumbering@quote</code> : <code>\initnumbering@sectcmd</code> becomes <code>\initnumbering@quote</code> . . . . .	294
<code>\l@advance@parledgroup@beforenormalnotes</code> : Some conde of <code>\l@dumboxmpfoot</code> moved to <code>\l@advance@parledegrou@beforenormalnotes</code> . . . . .	253
<code>\newseries@</code> : One endnotes file by series. . . . .	214
v2.0.1.	
General: Fix bug in <code>eledmac-compat</code> option . . . . .	1
Fix incompatibility between optional argument of <code>\pstart</code> and <code>\numberpstarttrue</code> . . . . .	1
v2.1.0.	
General: Fix bug with <code>\advance</code> at the beginning of a <code>\pstart</code> . . . . .	1
Fix bug with <code>\chapter</code> in optional argument of <code>\pstart</code> in parallel typesetting with <code>scrbook</code> . . . . .	1
Fix bug with <code>\eledchapter</code> in parallel typesetting with <code>scrbook</code> . . . . .	1
Fix bug with <code>\setline</code> at the beginning of a <code>\pstart</code> . . . . .	1
Fix spacing bug with <code>\Xhooknote</code> and <code>\bhooknoteX</code> when using them to insert text and not to execute code. . . . .	1



New tools to number stanzas . . . . .	1
v2.1.1.	
General: Fix bug with <code>\ledpbsetting{before}</code> . . . . .	1
v2.1.2.	
General: Fix bug with lineation by <code>pstart</code> and tabular environments (added in 2.1.0). . . . .	1
v2.1.3.	
General: <code>\Xhangindent</code> and <code>\hangindentX</code> work now with all the paragraphs in the note. . . . .	1
<code>\Xnoindent</code> and <code>\noindentX</code> work now again (broken in 2.0.0). . . . .	1
Change some internal code in order to provide compatibility with $\text{\LaTeX}$ release of October 2015 . . . . .	1
Fix bug which inserted double space before paragraphed familiar notes. . . . .	1
Fix bug with <code>\edindex</code> when using not-Latin characters without UTF-8 engines . . . . .	1
<code>\ledsetnormalparstuffX</code> : Replaced <code>\noindent</code> with <code>\parindent</code> set to 0pt. . . . .	194
v2.2.0.	
General: Fix bug with combination of <code>\onehalfspacing</code> and two columns and three columns notes typeset. . . . .	1
Fix bug with some setting command and optimization option. . . . .	1
Fix spurious space with paragraphed critical notes when using $\text{\Lua\LaTeX}$ . . . . .	1
Increase line list version number to ensure compatibility with new options of <code>reledpar</code> package. . . . .	1
New setting tools for endnotes: <code>\Xendnumberonlyfirstinline</code> , <code>\Xendnumberonlyfirstintwolines</code> , <code>\Xendsymmlinenumber</code> , <code>\Xendbeforenumber</code> , <code>\Xendafternumber</code> , <code>\Xendbeforemysymmlinenumber</code> , <code>\Xendaftersymmlinenumber</code> , <code>\Xendboxsymmlinenumber</code> , <code>\Xendhangindent</code> , <code>\Xendbhooklinenumber</code> , <code>\Xendahooklinenumber</code> , <code>\Xendbhookinplaceofnumber</code> , <code>\Xendahookinplaceofnumber</code> . . . . .	1
v2.2.1.	
General: Compatibility with $\text{\LaTeX}$ format 2015/10/01. . . . .	1
v2.2.2.	
General: Fix bug in <code>\sethangingsymbol</code> . . . . .	1
Fix bug with old version of <code>etex</code> . . . . .	1
v2.3.0.	
General: Disable empty lines as paragraph in stanza. . . . .	1
Fix compatibility of paragraphed footnotes with <code>bidi</code> v17.9 and following. . . . .	1
Warning message when using some setting commands inside <code>rightside</code> environment (deprecated behavior) . . . . .	1
v2.3.1.	
General: Fix spurious space when using optional argument of <code>\stanza</code> (introduced in v2.3.0). . . . .	1
v2.4.0.	
General: <code>\Xbhooknote</code> and <code>\bhooknoteX</code> work with notes in columns. . . . .	1
Fix bug of <code>\parindentX</code> and <code>\Xparindent</code> with two columns and three columns notes. . . . .	1
Fix bug with <code>\sameword</code> in right side. . . . .	1
Fix spurious space in two columns and three columns notes. . . . .	1
Fix spurious space when using optional argument of <code>stanza</code> (introduced in v2.3.0). . . . .	1
New hooks: <code>\Xlinerangeseparator</code> and <code>\Xendlinerangeseparator</code> . . . . .	31
Option <code>linerangesep</code> for critical footnotes and endnotes. . . . .	31
<code>\footnoteoptions@</code> : First argument of <code>\footnoteoption@</code> is now mandatory, not optional. . . . .	144

v2.4.1.	
General: Fix bug with <code>\appref</code> and <code>\apprefwithpage</code> (introduced in v2.4.0).	1
Fix bug with tabular environments when using <code>babel</code> or <code>polyglossia</code> languages that override $\TeX$ <code>\roman</code> command, like Greek language.	1
Fix bug with tabular environments when using <code>babel</code> or <code>polyglossia</code> languages that override $\TeX$ <code>\roman</code> command, like Greek.	1
v2.5.0.	
General: <code>\apprefwithpage</code> and <code>\appref</code> print double quotation mark when the label was not defined.	1
<code>\apprefwithpage</code> and <code>\appref</code> work with right side crossref.	1
<code>\apprefwithpage</code> works also when <code>noend</code> option is enabled.	1
<code>\appref</code> and <code>\apprefwithpage</code> can take <code>linrangesep</code> optional argument.	1
<code>\edlabel</code> works now in $\TeX$ <code>\footnote</code> .	1
<code>\lemma</code> can be used even when the <code>nocritical</code> is enabled.	1
Compatibility with new hook and tools of <code>reledpar</code> 2.6.0.	1
Fix spurious vertical space in <code>astanza</code> environment ( <code>reledpar</code> ).	1
Log now states ‘There were undefined references’ when using wrong references in <code>\edlineref</code> or <code>edpageref</code> .	1
New hooks to customize page and line number appearance in endnotes.	1
New hooks: <code>\Xbhookgroup</code> and <code>\bhookgroupX</code> .	1
New tools to easily make cross-reference to a passage defined by a start and an end line	47
<code>\edlabel</code> : Fix bug when calling <code>\edlabel</code> in a footnotes of the rightside	232
<code>\l@d@section</code> : <code>\endnotes</code> take six arguments.	200
<code>\printlines</code> : <code>\printlines</code> takes an eighth argument: the line flag	173
<code>\SErefwithpage</code> : Debug <code>\setapprefprefixsingle</code>	241
<code>\xlineref</code> : <code>\xlineref</code> does not include anymore the side flag. Use <code>\xflagref</code> to get it. Not that <code>\edlineref</code> still contains the flag.	236
v2.6.0.	
General: Adds compatibility with <code>innnote</code> and <code>notenumber</code> options of <code>indextools</code> package.	1
Fix bug with footnote counter in <code>ledgroup</code> (added in v2.5.0).	1
Fix bug, introduced in v2.5.0, with footnote numbering in parallel typesetting when using <code>perpage</code> package.	1
v2.7.0.	
<code>\@k</code> : <code>\rigidbalance</code> is split in <code>\Xrigidbalance</code> and <code>\rigidbalanceX</code> .	159
General: Add dash as default page range separator for <code>\SEonlypage</code>	1
Debug <code>\SEonlypage</code> when referring to only one page.	1
Delete parenthesis after <code>\SEonlypage</code> .	1
Fix (again) bugs with footnote numbering in parallel typesetting while using <code>ledgroup</code> environments (bug added in v2.5.0).	1
Fix bug with <code>\SErefwithpage</code> .	1
Fix bugs in compatibility with <code>innote</code> and <code>notenumber</code> options of <code>indextools</code> package, when indexing outside of a <code>ledgroup</code> .	1
New commands to make glossaries connected to page and linenumber with the <code>glossaries</code> package	1
New hooks: <code>\Xhsize</code> and <code>\hsizeX</code>	40
New hooks: <code>\Xlemmafnt</code> and <code>\Xendlemmafnt</code>	36
New setting commands: <code>\setSEonlypageprefixsingle</code> and <code>\setSEonlypageprefixmore</code>	1
Warning for duplicate and undefined labels are parsable by <code>latexmk</code>	1

Warning for duplicate labels does not send any more a false line and page number . . .	1
When using <code>hyperref</code> package, add link in familiar footnotes between the footnote marks in the text and the footnote marks in the footnote . . . . .	1
When using <code>hyperref</code> package, add links for <code>\SEref</code> and related, <code>\appref</code> and related. . . . .	1
When using <code>hyperref</code> package, add links from critical footnotes and critical endnotes to the line of text they refers . . . . .	1
<code>\l@do@section: \endnotes</code> take seven arguments. . . . .	200
v2.7.1.	
General: Debug <code>\Xhookgroup</code> hooks executed on columnar footnotes (moved to a larger group, to take effect). . . . .	1
v2.7.2.	
General: <code>\Xhsize</code> and <code>\hsizeX</code> become <code>\Xwidth</code> and <code>\widthX</code> . . . . .	40
Fix problem of hyphenation when using <code>hyperref</code> package (added in v2.7.0). . . . .	1
v2.8.0.	
General: <code>reledmac</code> cross-referencing can take advantage of <code>xr</code> package. . . . .	1
More <code>\edgls...</code> commands. . . . .	1
No indentation for paragraphed notes in <code>ledgroup</code> . Can be changed with <code>\Xparindent</code> and <code>\parindentX</code> . . . . .	1
<code>\l@do@section: \Xendhangindent</code> and <code>\Xendafternote</code> can take values which are relative to the font size of the endnote. . . . .	200
v2.8.1.	
General: Warning for undefined labels are really parsable by <code>latexmk</code> . . . . .	1
v2.8.2.	
General: Fix bug concerning indent in a paragraph immediatly following a sectioning command (bug NOT fixed on <code>reledpar</code> ) . . . . .	1
Fix bug with <code>\AtEveryPstart</code> added in version 2.0.0. . . . .	1
Fix bug with vertical space after between sectioning command as optional argument of a <code>\pstart</code> and <code>\pstart</code> content . . . . .	1
v2.9.0.	
General: Allow to make continuing line numbering between normal text and parallel text, using <code>\pausenumbering</code> and <code>\resumenumbering</code> and the <code>continuousnumberingwithcolumns</code> option. . . . .	1
Fix bug when using <code>\linenum{page}</code> and <code>\pausenumbering... \resumenumbering</code> . . . . .	1
Fix bug with three and two columns footnote setting (added in v.2.4.0). . . . .	1
Fix spurious space inside three columns familiar footnote. . . . .	1
Write correct metadata in numbered files when using <code>\pausenumbering... \resumenumbering</code> . . . . .	1