

# 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 . . . . .	12
1.2.3 eledmac . . . . .	13

---

\*This file (**reledmac.dtx**) has version number v2.7.0, last revised 2015/11/29.

<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 . . . . .	14
<b>4 Text lines and paragraphs numbering</b>	<b>15</b>
4.1 Text lines numbering . . . . .	15
4.2 Paragraphs . . . . .	15
4.2.1 Basics . . . . .	15
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 . . . . .	17
4.2.7 Memory limits . . . . .	17
4.3 Lineation commands . . . . .	18
4.3.1 Disabling lineation . . . . .	18
4.3.2 Setting lineation start and step . . . . .	18
4.3.3 Setting lineation reset . . . . .	19
4.3.4 Setting line number margin . . . . .	19
4.3.5 Other settings . . . . .	19
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>21</b>
5.1 Terminology . . . . .	21
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 . . . . .	25
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 . . . . .	26

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 . . . . .	28
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>29</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 . . . . .	33
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
<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 . . . . .	50
11.2.4 Separator between notes . . . . .	51

<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>53</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 . . . . .	58
<b>18 Miscellaneous</b>	<b>59</b>
18.1 Known and suspected limitations . . . . .	59
18.1.1 ‘No room for a new’ . . . . .	59
18.1.2 Marginal notes . . . . .	60
18.1.3 Paragraph shape . . . . .	60
18.1.4 Paragraphed footnotes . . . . .	60
18.1.5 Use with other packages . . . . .	61
18.1.6 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 . . . . .	72
II.10 Prepare reledpar . . . . .	73

II.11 Booleans provided by other optional packages which are required in any case . . . . .	73
<b>III Sectioning commands</b>	<b>74</b>
<b>IV List macros</b>	<b>77</b>
<b>V Line counting</b>	<b>79</b>
V.1 Choosing the system of lineation . . . . .	79
V.2 Line number margin . . . . .	80
V.3 Line number initialization and increment . . . . .	81
V.4 Line number locking . . . . .	83
V.5 Line number style . . . . .	84
V.6 Line number printing . . . . .	84
V.7 Line number counters and lists . . . . .	85
V.8 Line number locking counter . . . . .	86
V.9 Line number associated to lemma . . . . .	86
V.10 Reading the line-list file . . . . .	89
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 . . . . .	115
VI.3 Substitute line numbers . . . . .	116
VI.4 Lemma disambiguation . . . . .	117
<b>VII Paragraph decomposition and reassembly</b>	<b>123</b>
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	123
VII.2 Processing one line . . . . .	128
VII.2.1 General process . . . . .	128
VII.2.2 Process for “normal” line . . . . .	129
VII.2.3 Process for line containing <code>\eledsection</code> command . . . . .	130
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 . . . . .	140
XI.2 Penalties . . . . .	141
XI.3 Printing leftover notes . . . . .	142

<b>XII Critical footnotes</b>	<b>143</b>
XII.1 Fonts . . . . .	143
XII.2 Individual note options . . . . .	143
XII.3 Notes language . . . . .	144
XII.4 General survey of the way we manage notes . . . . .	145
XII.5 General setup . . . . .	145
XII.6 Footnotes arrangement . . . . .	146
XII.6.1 User level macro . . . . .	146
XII.6.2 Normal footnote . . . . .	147
XII.6.3 Paragraphed footnotes . . . . .	151
XII.6.4 Columnar footnotes . . . . .	158
XII.7 Critical notes presentation . . . . .	164
XII.7.1 Font tools . . . . .	164
XII.7.2 Pstart number in footnote . . . . .	165
XII.7.3 Line number printing . . . . .	165
<b>XIII Familiar footnotes</b>	<b>174</b>
XIII.1 Adjacent footnotes . . . . .	174
XIII.2 Regular footnotes for numbered texts . . . . .	175
XIII.3 Footnote formats . . . . .	177
XIII.4 Footnote arrangement . . . . .	178
XIII.4.1 User level macro . . . . .	178
XIII.4.2 Normal footnotes . . . . .	178
XIII.4.3 Two columns footnotes . . . . .	184
XIII.4.4 Three columns footnotes . . . . .	186
XIII.4.5 Paragraphed footnotes . . . . .	188
XIII.5 Wrapping footnote marks in hyperlink . . . . .	192
<b>XIV Code common to both critical and familiar footnote in normal arrangement</b>	<b>193</b>
<b>XV Footnotes' width for two columns</b>	<b>194</b>
<b>XVI Footnotes' order</b>	<b>195</b>
<b>XVII Footnotes' rule</b>	<b>195</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 . . . . .	196
XVIII.0.3 Internal commands . . . . .	197
<b>XIX Endnotes</b>	<b>198</b>

<b>XX Generate series of notes</b>	<b>206</b>
XX.1 Test if series is still existing . . . . .	207
XX.2 Init specific to <code>reledpar</code> . . . . .	207
XX.3 For critical footnotes . . . . .	207
XX.3.1 Options . . . . .	207
XX.3.2 Create inserts, needed to add notes in foot . . . . .	208
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc. . . . .	208
XX.3.4 Set standard display . . . . .	211
XX.4 For familiar footnotes . . . . .	211
XX.4.1 Options . . . . .	211
XX.4.2 Create tools for familiar footnotes ( <code>\footnotex</code> ) . . . . .	212
XX.5 The endnotes . . . . .	213
XX.5.1 The auxiliary file . . . . .	213
XX.5.2 The main macro . . . . .	213
XX.5.3 The options . . . . .	214
XX.6 Init standards series (A,B,C,D,E) . . . . .	216
<b>XXI Setting series display</b>	<b>216</b>
XXI.1 Change series order . . . . .	216
XXI.2 Test series order . . . . .	216
XXI.2.1 Get the first series . . . . .	217
XXI.3 Series setting . . . . .	217
XXI.3.1 General way of working . . . . .	217
XXI.3.2 Tools to set options . . . . .	217
XXI.3.3 Tools to generate options commands . . . . .	219
XXI.3.4 Options for critical notes . . . . .	220
XXI.3.5 Options for familiar notes . . . . .	222
XXI.3.6 Options for endnotes . . . . .	222
XXI.4 Hooks for a particular footnote . . . . .	224
XXI.5 Alias . . . . .	225
<b>XXII Output routine</b>	<b>225</b>
XXII.0.1 Page number management . . . . .	225
XXII.0.2 Extra footnotes output . . . . .	225
XXII.0.3 Standard output's commands patching . . . . .	228
<b>XXIII Cross referencing</b>	<b>230</b>
<b>XXIV Side notes</b>	<b>243</b>
<b>XXV Minipages and such</b>	<b>249</b>



<b>XXVI Indexing</b>	<b>254</b>
XXVI.1 Looking on package order . . . . .	254
XXVI.2 Auxiliary macros for <code>\edindex</code> . . . . .	254
XXVI.3 Code specific to <code>\edindexin</code> critical footnotes . . . . .	255
XXVI.4 Analysis of command in indexed text . . . . .	256
XXVI.5 Code for the formatted index . . . . .	257
XXVI.6 Main code . . . . .	257
XXVI.7 Hyperlink . . . . .	258
XXVI.8 ‘innote’ and ‘notenumber’ option of <code>indextools</code> package . . . . .	261
<b>XXVII Glossaries</b>	<b>262</b>
<b>XXVIII Verse</b>	<b>263</b>
XXVIII.1 Hanging symbol management . . . . .	263
XXVIII.2 Using <code>&amp;</code> character . . . . .	263
XXVIII.3 Code category setting . . . . .	264
XXVIII.4 Stanza count and indent . . . . .	264
XXVIII.5 Numbering stanza . . . . .	265
XXVIII.6 Stanza number in note . . . . .	266
XXVIII.7 Main work . . . . .	267
XXVIII.8 Restore catcode and penalties . . . . .	269
<b>XXIX Arrays and tables</b>	<b>269</b>
XXIX.1 Preamble: macro as environment . . . . .	269
XXIX.2 Tabular environments . . . . .	272
XXIX.2.1 Disabling and restoring commands . . . . .	273
XXIX.2.2 Counters, boxes and lengths . . . . .	276
XXIX.2.3 Tabular typesetting . . . . .	280
XXIX.2.4 Environments . . . . .	291
<b>XXX Quotation’s commands</b>	<b>291</b>
<b>XXXI Section’s title commands</b>	<b>292</b>
XXXI.1 Commands to disable some feature . . . . .	292
XXXI.2 General overview . . . . .	293
XXXI.3 <code>\beforeeledchapter</code> command . . . . .	293
XXXI.4 Auxiliary commands . . . . .	294
XXXI.5 Patching standard commands . . . . .	295
XXXI.6 Main code of <code>\eledxxx</code> commands . . . . .	299
XXXI.7 Macros written in the auxiliary file . . . . .	302
<b>XXXII Page breaking or no page breaking depending of specific lines</b>	<b>304</b>
<b>XXXIII Long verse: prevents being separated by a page break</b>	<b>306</b>
<b>XXXIV Compatibility with <code>eledmac</code></b>	<b>306</b>

<b>Appendix A Some things to do when changing version</b>	<b>309</b>
Appendix A.1 Migrating from edmac to ledmac . . . . .	309
Appendix A.2 Migration from ledmac to eledmac . . . . .	310
Appendix A.3 Migration to eledmac 1.5.1 . . . . .	311
Appendix A.4 Migration to eledmac 1.12.0 . . . . .	311
Appendix A.5 Migration to eledmac 17.1 . . . . .	312
Appendix A.6 Migration to eledmac 1.21.0 . . . . .	312
Appendix A.6.1 \Xledsetnormalparstuffand\ledsetnormalparstuffX	312
Appendix A.6.2 Endnotes . . . . .	312
Appendix A.7 Migration to eledmac 1.22.0 . . . . .	312
Appendix A.8 Migration to eledmac 1.23.0 . . . . .	312
Appendix A.9 Migration from eledmac to reledmac . . . . .	313
Appendix A.9.1 Risk of ‘no room for a new’ . . . . .	313
Appendix A.9.2 Multiple indices with memoir . . . . .	313
Appendix A.9.3 Deprecated commands and options . . . . .	313
Appendix A.9.4 \renewcommandreplaced by command . . . . .	314
Appendix A.9.5 Commands the names of which have been changed . . . .	314
Appendix A.9.6 Endnotes . . . . .	316
Appendix A.9.7 Z Series . . . . .	316
Appendix A.9.8 Internal commands . . . . .	316
Appendix A.10 Migration to reledmac 2.1.0 . . . . .	316
Appendix A.11 Migration to reledmac 2.1.3 . . . . .	316
Appendix A.12 Migration to reledmac 2.3.0 . . . . .	316
Appendix A.13 Migration to reledmac 2.4.0 . . . . .	317
Appendix A.14 Migration to reledmac 2.5.0 . . . . .	317
Appendix A.15 Migration to reledmac 2.7.0 . . . . .	317
<b>References</b>	<b>318</b>
<b>Index</b>	<b>318</b>
<b>Change History</b>	<b>362</b>

## 1 Introduction

### 1.1 Aim of the package

The `reledmac` package, together with  $\text{\LaTeX}$ , 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.  $\text{\LaTeX}$  and `Eledmac` will take care of the formatting and visual correlation of all the disparate types of information.

Apart from `reledmac` there are other  $\text{\LaTeX}$  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 de-

<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).

scription 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 Quinqueecclesiensi 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.

### 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.

<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: Schipphower 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. Gädke 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.

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. 29). 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. 310.

### 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. 313.

## 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 L<sup>A</sup>T<sub>E</sub>X once or twice more.

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. 314).

**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 18.1.6 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`)

`\numberpstarttrue` It is possible to insert a number at every `\pstart` command; you must use the  
`\numberpstartfalse` `\numberpstarttrue` command to have it. You can stop the numbering with `\numberpstartfalse`.  
`\thepstart` You can redefine the command `\thepstart` to change style. You can change the value  
of the `pstart` number by using *after* `\beginnumbering`:

```
\setcounter{numberpstart}{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`.

### 4.2.6 Languages written in Right to Left

If you use languages written right to left with `LuaLaTeX` or `XLaTeX`, 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 `LaTeX`.**

`\pausenumbering` `reledmac` stores a lot of information about line numbers and footnotes in memory  
`\resumenumbering` as it goes through a numbered section. But at the end of such a section, it empties its

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

memory out, so to speak. If your text has a very long numbered section it is possible that your  $\text{\LaTeX}$  may reach its memory limit. There are two solutions to this.

The first solution is to get a larger  $\text{\LaTeX}$  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` Line numbering can be disabled with `\numberlinefalse`. It can be enabled again with `\numberlinetrue`.

#### 4.3.2 Setting lineation start and step

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

`\sublinenumincrement` You can define `\linenumberlist` to specify a non-uniform distribution of printed

`\linenumberlist`

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` `\rightlinenum` `\linenumsep` When a marginal line number is to be printed, there are many ways to display it. You can redefine `\leftlinenum` and `\rightlinenum` to change the way marginal line numbers 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 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 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 just 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 calculated: 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 :		1 I am happy : I saw my friend Smith on
I saw my friend <code>\edtext{Smith}{</code>		2 Tuesday.
<code>\Afootnote{Jones C, D.}}</code>		
on 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 : <code>\edtext{I saw my friend</code>	1 I am happy : I saw my friend Smith on
<code>\edtext{Smith}{\Afootnote{Jones</code>	2 Tuesday.
<code>C, D.}}</code> on Tuesday.}{	
<code>\Bfootnote{The date was</code>	1 Smith] Jones C, D.
<code>July 16, 1954.}</code>	
}	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,  $\langle commands \rangle$ , 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` to disable line numbering for this note.
- `nosep` to disable the lemma separator for this note.
- `linangesep=c` to change to  $\langle c \rangle$  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}`.

$\langle option \rangle$  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.
- `linangesep=c` to change to  $\langle c \rangle$  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  
 $\langle s \rangle$  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  $\langle s \rangle$  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}}
\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.

<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`.

### 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  $\text{\XeTeX}$  or  $\text{\LuaTeX}$ , 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, “ $\alpha$ ” 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  $\text{\XeTeX}$ , add this line in your preamble: `\XeTeXinputnormalization 1`.
- If you use  $\text{\LuaTeX}$ , use the `uninormalize` package of Michal Hoftich<sup>15</sup> with the `buffer` option set to true.

With these tools,  $\text{\XeTeX}$  /  $\text{\LuaTeX}$  will dynamicaly 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
```

---

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

```

\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`.

The diagram illustrates nested `\edtext` levels. The outer box represents level 2 and contains the text "1<sub>inlemma</sub>" followed by a smaller box representing level 1. This inner box contains "2" followed by "3<sub>2</sub>". To the right of the inner box is "1...3" followed by "4" and "5<sub>1</sub>". To the right of the outer box is "1...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  $\text{\LaTeX}$  footnotes generated via `\footnote`, the package also provides five series of additional footnotes called `\footnoteA` through `\footnoteE`. These have the familiar marker in the text, and the marked text at the foot of the page can be formatted using any of the styles described for the critical footnotes. Note that the ‘regular’ footnotes have the series letter at the end of the macro name whereas the critical 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 series A is defined by the `\thefootnoteA` macro; the default is:  
`\bodyfootmarkA` `\renewcommand*{\thefootnoteA}{\arabic{footnoteA}}`  
`\footfootmarkA` 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 `[<s>]`, 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 (`<l>`) is used, it can be stretchable: `a plus b minus c`. The final length `m` is calculated by  $\TeX$  to have  $a - c \leq m \leq a + b$  units. If you use some relative unit<sup>16</sup>, it will be relative to the font size of the footnote, except for commands concerning

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

the place kept by the notes, including blank spaces.

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  $\text{\TeX}$  to have:  $a - c \leq m \leq a + b$ . If you use some relative unit<sup>17</sup>, it will be relative to font size of the footnote, except for commands concerning the place kept by the notes — including blank space.

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 18.1.4 p. 60.
- `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.

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.

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

---

<sup>17</sup>Like `em` which is the width of an ‘m’ in a given font.

## 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⟩]`.

Use `\Xnumberonlyfirstinline[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

`\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. Use the command `\Xnumberonlyfirstintwolines[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

`\Xsymlinenum` For setting a particular symbol in place of the line number, you can use `\Xsymlinenum[⟨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`  
`\Xendnumberonlyfirstintwolines` and `\Xendsymlinenum` are the equivalents of  
`\Xendsymlinenum` `\Xnumberonlyfirstinline`; `\Xnumberonlyfirstintwolines` and `\Xsymlinenum`.

### 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

`\Xlinrangeseparator` 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 `\Xlinrangeseparator[⟨s⟩]{⟨text⟩}`, and with `\Xendlinrangeseparator[⟨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 line numbers. Instead of this, it is also possible to print an abbreviation which stands for "line 1 and subsequent line(s)".  
`\Xmorethantwolines`

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  $\langle text \rangle$  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[ $\langle series \rangle$ ]`.

It is possible to disable this again with `\Xtwolinesbutnotmore[ $\langle series \rangle$ ][false]`.

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[ $\langle series \rangle$ ]`.

Use `\Xtwolinesonlyinsamepage[ $\langle series \rangle$ ][false]` to disable this.

You can disable `\Xtwolines` and related for a specific note by using the ‘[fulllines]’ 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[ $\langle s \rangle$ ]` if you do not want to have the line number in a footnote. To cancel it, use `\Xnonumber[ $\langle s \rangle$ ][false]`. `\Xendnonumber[ $\langle s \rangle$ ]` is the same for endnote.

### 6.2.6 Printing pstart number

`\Xpstart`

You can use `\Xpstart[ $\langle s \rangle$ ]` if you want to print the pstart number in the footnote, before the line and subline number. Use `\Xpstart[ $\langle s \rangle$ ][false]` to disable this.  $\langle s \rangle$  can be empty if you want to disable it for every series. 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[ $\langle s \rangle$ ]`. In this case, the pstart number will be printed every time in footnote.

`\Xonlypstart`

In combination with `\Xpstart`, you can use `\Xonlypstart[ $\langle s \rangle$ ]` if you want to print only the pstart number in the footnote, and not the line and subline number. Use `\Xonlypstart[ $\langle s \rangle$ ][false]` disable this.  $\langle s \rangle$  can be empty, if you want to disable it for every series.



### 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. Use `\Xstanza[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

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

`\Xnonbreakableafternumber` By default, the space defined by `\Xafternumber` is breakable. With `\Xnonbreakableafternumber[⟨s⟩]` it becomes nonbreakable. Use `\Xnonbreakableafternumber[⟨s⟩][false]` to disable this. `⟨s⟩` can be empty if you want to disable it for every series.

### 6.2.10 Space around line symbol

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

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

`\Xendbeforemylinenum` `\Xendbeforemylinenum` and `\Xendaftersymylinenum` are the equivalents of `\Xbeforemylinenum` and `\Xaftersymylinenum` for the endnotes.

### 6.2.11 Space in place of number

`\Xinplaceofnumber` If no number or symbolic line number is printed, you can add a space, with `\Xinplaceofnumber[⟨s⟩]{⟨l⟩}`.

The default value is 1 em.

`\Xendinplaceofnumber` `\Xendinplaceofnumber[⟨s⟩]{⟨l⟩}` is the same, for critical endnotes.

### 6.2.12 Boxing line number and line symbol

`\Xboxlinenum` 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 `\Xboxlinenum[⟨s⟩]{⟨l⟩}` to do that. To subsequently disable this feature, use `\Xboxlinenum` 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:

```
\Xhangindent{1em}
\Xafternumber{0em}
\Xboxlinenum{1em}
```

`\Xboxsymlinenum` `\Xboxsymlinenum[⟨s⟩]{⟨l⟩}` is the same as `\Xboxlinenum` but for the line number symbol.

`\Xendboxsymlinenum` `\Xendboxsymlinenum[⟨s⟩]{⟨l⟩}` is the same as `\Xboxsymlinenum` but for endnotes.

`\Xboxlinenumalign` If you put line number in box, it will be aligned left inside the box. However, you can change it using `\Xboxlinenumalign[⟨s⟩]{⟨text⟩}` where `⟨text⟩` can be the following:

**L** to align left (default value);

**R** to align right;

**C** 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.
```

`\Xendboxlinenum` `\Xendboxlinenum[⟨s⟩]{⟨l⟩}`, `\Xendboxlinenumalign[⟨s⟩]{⟨text⟩}`, `\Xendboxstartlinenum[⟨s⟩]{⟨l⟩}`

`\Xendboxlinenumalign` `\Xendboxendlinenum[⟨s⟩]{⟨l⟩}` are the same as, respectively, `\Xboxlinenum` and

`\Xendboxstartlinenumalign` `\Xboxlinenumalign`, `\Xboxstartlinenum`, `\Xboxendlinenum` except in endnotes.

`\Xendboxendlinenumalign`

### 6.3 For endnotes

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

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

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

`\Xendlineprefixmore` `\Xendlineprefixmore[\langle s \rangle]{\langle text \rangle}` 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[\langle s \rangle]{\langle code \rangle}` 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[\langle s \rangle]{\langle code \rangle}` is used to execute code after line number in endnotes. The code is executed after the `\Xendafternumber` space.

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

`\Xendahookinplaceofnumber` `\Xendahookinplaceofnumber[\langle s \rangle]{\langle code \rangle}` is used to execute code after space or symbol which replace line number in endnotes. The code is executed after the `\Xendaftersymlinenum` 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>18</sup>. You can use `\Xlemmaseparator[\langle s \rangle]{\langle Xlemmaseparator \rangle}` 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[\langle s \rangle]{\langle l \rangle}` 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[\langle s \rangle]{\langle l \rangle}` 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[\langle s \rangle]`, which is simply a alias of `\Xlemmaseparator[\langle s \rangle]{}`.

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

<sup>18</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 \hspace</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 \hspace</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 \hspace</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 \hspace</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{\$\parallel\$}</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[ $\langle s \rangle$ ]{ $\langle text \rangle$ }`.

### 6.11.2 Code before notes

`\Xbhookgroup` While `\Xtxtbeforenotes` is for typesetting code before notes, `\Xbhookgroup` and `\bhookgroupX` (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[ $\langle s \rangle$ ]{ $\langle l \rangle$ }`. The default value is 1.2em plus .6em minus .6em.

**Be careful, the standard  $\LaTeX$  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[ $\langle s \rangle$ ]{ $\langle l \rangle$ }`. The default value is 1.2em plus .6em minus .6em.

**Be careful, the standard  $\LaTeX$  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{ $\langle l \rangle$ }`. 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{ $\langle l \rangle$ }`. 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[ $\langle s \rangle$ ]{ $\langle l \rangle$ }`. The default value is 0pt.

**Be careful, the standard  $\LaTeX$  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[ $\langle s \rangle$ ]{ $\langle l \rangle$ }`. The default value is 0pt.

**Be careful, the standard  $\LaTeX$  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[ $\langle s \rangle$ ]{ $\langle l \rangle$ }`. 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

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

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

```
\Xhsize{\columnwidth+\marginparsep+\ledrsnotewidth}
\hsizeX{\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. Use `\Xnoteswidthliketwocolumns[⟨s⟩][false]` to disable it.
- `\noteswidthliketwocolumnsX[⟨s⟩]` to create familiar notes with a two-column size width. Use `\noteswidthliketwocolumnsX[⟨s⟩][false]` to disable it.

## 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@lemmafont` We will briefly discuss `\select@lemmafont` 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@lemmafont` 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@lemmafont` 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@lemmafont` selects the appropriate font for the note using that font specifier.

`reledmac` uses `\select@lemmafont` 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.

`\setstanzaindents` 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. 311.**

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` 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` 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. 58 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.

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

`\begin{ledgroupsize}[\langle pos \rangle]{\langle width \rangle}`.

The required `\langle width \rangle` argument is the text width for the environment. The optional `\langle pos \rangle` 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{ledgroupsize}{\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{\langle lab \rangle}`. `\langle lab \rangle` 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>19</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  $\TeX$ .aux file. You will need to process your document through  $\TeX$  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  $\TeX$  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. 24).

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>19</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. 24 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 $\TeX$ 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. 24) to refer to these labels, `reledmac` will use your line settings to refer to the passage.

`\appref`      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

`\apprefwithpage`      the second will print the lines as they are printed in endnotes.

### 10.6.3 Settings

`\setapprefprefixsingle`      **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`      `\setSerefprefixsingle` and `\setSerefprefixmore` are similar for `\Seref` command.

`\setSerefonlypageprefixsingle`      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).

## 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{⟨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}`.

`\ledrightnote`      The package's default setting is

`\ledsidenote`      `\sidenotemargin{right}`

`\sidenotemargin`      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`.

`\ledrsnotewidth`

#### 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).

`\leftnoteupfalse`

#### 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`      These macros specify how the sidenote texts are to be typeset. The initial definitions are:

`\ledlsnotefontsetup`

`\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

`\setsidenotessep` 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 `\setsidenotessep{<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 & line number 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 & lineNumber 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>20</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>21</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:

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>22</sup>.

<sup>20</sup>For further detail, you can read <http://tex.stackexchange.com/a/32783/7712>.

<sup>21</sup>Or, for people who read French, read <http://geekographie.maieul.net/174>.

<sup>22</sup>These are the recommended lines to provide the best possible compatibility between `hyperref` and xindy, even without using `reledmac`.

## 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 Preamble 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, there will be 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 arguments are the same as for the standard commands. So for example:

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

## 14 Tabular material

$\TeX$ '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.

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

```
edarrayl
edarrayc
edarrayr
edtabularl
edtabularc
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 `\hline`). 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{edtabularl}
\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 `{<math>}` but the `{<math>}` has no effect on the  
`\spreadtext` calculation of column widths. `\spreadtext{<text>}` is the analagous 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 `<start>` to `<end>` inclusive with `<fill>`. The `<fill>` 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          & 2    & 3    & 4    & 5  \\
Q          &      & fd   & h    & qwertziohg \\
v          & wptz & x    & y    & vb  \\
g          & nnn  & \edrowfill{3}{5}{\upbracefill} & & \\
\edrowfill{1}{3}{\downbracefill} &      &      & pq   & dgh \\
k          &      & l    & co   & ghweropjklmnbvcxys \\
1          & 2    & 3    & \edrowfill{4}{5}{\hrulefill} & \\
\end{tabularr}
```

1	2	3	4	5
Q		fd	h	qwertziohg
v	wptz	x	y	vb
g	nnn	$\overbrace{\hspace{10em}}$		
k	$\underbrace{\hspace{10em}}$		pq	dgh
1	2	3	co	ghweropjklmnbvcxys
			$\underline{\hspace{10em}}$	

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          & & 3 & 4  \\
a & \edrowfill{2}{3}{\upbracketfill} & & d  \\
A & B          & & C & D  \\
\end{edarrayc}
```

1	2	3	4
$a$	$\lrcorner$		$d$
$A$	$B$	$C$	$D$

`\edatleft`      `\edatleft[\langle math \rangle]{\langle symbol \rangle}{\langle halfheight \rangle}` typesets the math  $\langle symbol \rangle$  as  $\left\langle symbol \right\}$  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  $\left\langle symbol \right\}$  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}

```

$$left = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = right$$

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

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}

```

	$\begin{matrix} A & 1 & 2 & 3 \\ B & 1 & 3 & 6 \\ C & 1 & 4 & 8 \\ D & 1 & 5 & 0 \end{matrix}$	$\begin{matrix} 8 \\ 0 \end{matrix}$
Before		After

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

```

\begin{edarrayr}
a & b & C & d & \\
v & w & x & y & \\
m & n & o & p & \\
k & & L & cvb & \edvertline{4pc}
\end{edarrayr}

```

$a$	$b$	$C$	$d$	
$v$	$w$	$x$	$y$	
$m$	$n$	$o$	$p$	
$k$		$L$	$cvb$	

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

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.  $n + 1$ .

`\ledpbsetting`

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

`\lednopbinversettrue`

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 `\lednopbinversettrue` 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 ‘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  $\TeX$  will be slower.

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

### 18.1.2 Marginal notes

In general, `reledmac`'s system for adding marginal line numbers breaks anything that makes direct use of the  $\TeX$  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.

### 18.1.3 Paragraph shape

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

`\ballast`  $\TeX$  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.

### 18.1.4 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.

### 18.1.5 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>23</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
\newcommand{\morenoexpands}{\let\colorbox\@secondoftwo}
\makeatother
```

(`\@secondoftwo` is an internal  $\TeX$  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.

<sup>23</sup>Reported by Dirk-Jan Dekker in the CTT thread 'Incompatibility of "color" package' on 2003/08/28.

### 18.1.6 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}[2015/11/29 v2.7.0 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
36 \newif\ifledfinal
37 \DeclareOptionX{final}{\ledfinaltrue}
38 \DeclareOptionX{draft}{\ledfinalfalse}

```



```

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

```

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.

```

69 \ProcessOptionsX*\relax
70
71 %

```

## 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.

```

72 \RequirePackage{xargs}
73 \RequirePackage{etoolbox}

```

```

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

```

## II.5 Compatibility with Lua<sub>T</sub><sub>E</sub>X

Here, we enable some primitives for Lua<sub>T</sub><sub>E</sub>X.

```

85 \ifx\directlua\undefined\else%
86 \directlua{tex.enableprimitives("",{"textdir","pdir","bodydir"})}
87 \fi
88 %

```

## II.6 Boolean flags

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

```

89 \newif\ifl@dmemoir
90 \@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
91
92 %

```

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

```

93 \newif\if@ledgroup%
94 %

```

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

```

95 \newif\ifl@imakeidx
96 \@ifpackageloaded{imakeidx}{\l@imakeidxtrue}{}%False is the default value
97 %

```

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

```

98 \newif\ifl@indextools%
99 \@ifpackageloaded{indextools}{%
100 \l@indextoolstrue%
101 \l@imakeidxtrue%
102 \let\imki@wrindexentry\indtl@wrindexentry%
103 }{}%
104 %

```

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.

```
105 \ifdef{\if@RTL}{-}{\newif\if@RTL}
106 %
```

## 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.

```
107 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
108 %
```

`\reledmac@error` Write an error message.

```
109 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
110 %
```

```
\led@err@NumberingStarted11 \newcommand*{\led@err@NumberingStarted}{%
d@err@NumberingNotStarted12 \reledmac@error{Numbering has already been started}{\@ehc}}
NumberingShouldHaveStarted13 \newcommand*{\led@err@NumberingNotStarted}{%
114 \reledmac@error{Numbering was not started}{\@ehc}}
115 \newcommand*{\led@err@NumberingShouldHaveStarted}{%
116 \reledmac@error{Numbering should already have been started}{\@ehc}}
117 %
```

```
d@err@edtextoutsidestart18 \newcommand*{\led@err@edtextoutsidestart}{%
119 \reledmac@error{\string\edtext\space outside numbered paragraph (\pstart\
ldots\pend)}{\@ehc}}%
120 %
```

```
\led@mess@NotesChanged21 \newcommand*{\led@mess@NotesChanged}{%
122 \typeout{reledmac reminder: }%
123 \typeout{ The number of the footnotes in this section
124 has changed since the last run.}%
125 \typeout{ You will need to run LaTeX two more times
126 before the footnote placement}%
127 \typeout{ and line numbering in this section are
128 correct.}}
129 %
```

```

\led@mess@SectionContinued30 \newcommand*{\led@mess@SectionContinued}[1]{%
131   \message{Section #1 (continuing the previous section)}
132   %

\led@err@LineationInNumbered33 \newcommand*{\led@err@LineationInNumbered}{%
134   \reledmac@error{You can't use \string\lineation\space within
135       a numbered section}{\@ehc}}
136   %

\led@warn@BadLineation37 \newcommand*{\led@warn@BadLineation}{%
\led@warn@BadLinenummargin38   \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp39 \newcommand*{\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp40   \reledmac@warning{Bad \string\linenummargin\space argument}}
141 \newcommand*{\led@warn@BadLockdisp}{%
142   \reledmac@warning{Bad \string\lockdisp\space argument}}
143 \newcommand*{\led@warn@BadSublockdisp}{%
144   \reledmac@warning{Bad \string\sublockdisp\space argument}}
145   %

\led@warn@NoLineFile46 \newcommand*{\led@warn@NoLineFile}[1]{%
147   \reledmac@warning{Can't find line-list file #1}}
148   %

\led@warn@LineFileObsolete49 \newcommand*{\led@warn@Obsolete}[1]{%
150   \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
151   %

\led@warn@BadAdvancelineSubline52 \newcommand*{\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine53   \reledmac@warning{\string\advanceline\space produced a sub-line
154       number less than zero.}}
155 \newcommand*{\led@warn@BadAdvancelineLine}{%
156   \reledmac@warning{\string\advanceline\space produced a line
157       number less than zero.}}
158   %

\led@warn@BadSetline59 \newcommand*{\led@warn@BadSetline}{%
\led@warn@BadSetlinenum60   \reledmac@warning{Bad \string\setline\space argument}}
161 \newcommand*{\led@warn@BadSetlinenum}{%
162   \reledmac@warning{Bad \string\setlinenum\space argument}}
163   %

```

```

\led@err@PstartNotNumbered\newcommand*{\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart\reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumberednumbered section}{\@ehc}}
\led@err@PendNoPstart\newcommand*{\led@err@PstartInPstart}{%
\reledmac@error{\string\pstart\space encountered while another
\string\pstart\space was in effect}{\@ehc}}
\led@err@AutoparNotNumbered\newcommand*{\led@err@PendNotNumbered}{%
\reledmac@error{\string\pend\space must be used within a
\err@NumberingWithoutPstartnumbered section}{\@ehc}}
\newcommand*{\led@err@PendNoPstart}{%
\reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
\newcommand*{\led@err@AutoparNotNumbered}{%
\reledmac@error{\string\autopar\space must be used within a
numbered section}{\@ehc}}
\newcommand*{\led@err@NumberingWithoutPstart}{%
\reledmac@error{\string\beginnumbering...\string\endnumbering\space
without \string\pstart}{\@ehc}}%
%

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

\led@warn@DuplicateLabel\newcommand*{\led@warn@DuplicateLabel}[1]{%
\reledmac@warning{Duplicate definition of label `#1'\@gobble}%
\led@warn@AppLabelOutEdtext\@latex@warning@no@line{Label `#1' multiply defined}%
\led@warn@RefUndefined}%
\newcommand*{\led@warn@AppLabelOutEdtext}[1]{%
\reledmac@warning{\string\applabel\space outside of \string\edtext\space
`#1' on page \the\pageno.}}%
\newcommand*{\led@warn@RefUndefined}[1]{%
\G@refundefinedtrue%
\reledmac@warning{Reference `#1' on page \the\pageno\space undefined.%
Using `000'.}%
\@latex@warning{Reference `#1' on page \thepage \space%
undefined}%
}%
\newcommand*{\led@warn@pairRefUndefined}[1]{%
\G@refundefinedtrue%
\reledmac@warning{Reference `#1:start' and/or `#1:end' on page \the\
pageno\space undefined.
Using `??'.}}
%

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

```

204 %

\led@warn@BadSidenotemargin 205 \newcommand*{\led@warn@BadSidenotemargin}{%
206 \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
207 %

\led@warn@NoIndexFile 208 \newcommand*{\led@warn@NoIndexFile}[1]{%
209 \reledmac@warning{Undefined index file #1}}
210 %

\led@warn@SeriesStillExist 211 \newcommand{\led@warn@SeriesStillExist}[1]{%
212 \reledmac@warning{Series #1 is still existing !}}%
213 }%
214 %

\led@err@ManySidenotes 215 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes 216 \ifledRcol{%
\led@err@ManyRightnotes 217 \reledmac@warning{\itemcount@\space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
218 \else%
219 \reledmac@warning{\itemcount@\space sidenotes on line \the\line@num\
space p. \the\page@num}%
220 \fi%
221 }%
222 \newcommand{\led@err@ManyLeftnotes}{%
223 \ifledRcol{%
224 \reledmac@warning{\itemcount@\space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
225 \else%
226 \reledmac@warning{\itemcount@\space leftnotes on line \the\line@num\
space p. \the\page@num}%
227 \fi%
228 }%
229 \newcommand{\led@err@ManyRightnotes}{%
230 \ifledRcol{%
231 \reledmac@warning{\itemcount@\space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
232 \else%
233 \reledmac@warning{\itemcount@\space rightnotes on line \the\line@num\
space p. \the\page@num}%
234 \fi%
235 }%
236 %

```

```

\led@err@TooManyColumns37 \newcommand*\led@err@TooManyColumns}{%
\led@err@UnequalColumns38 \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn39 \newcommand*\led@err@UnequalColumns}{%
\led@err@HighEndColumn40 \reledmac@error{Number of columns is not equal to the number
\led@err@ReverseColumns41 in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
242 \newcommand*\led@err@LowStartColumn}{%
243 \reledmac@error{Start column is too low}{\@ehc}}
244 \newcommand*\led@err@HighEndColumn}{%
245 \reledmac@error{End column is too high}{\@ehc}}
246 \newcommand*\led@err@ReverseColumns}{%
247 \reledmac@error{Start column is greater than end column}{\@ehc}}
248 %

err@EdtextWithoutFootnote49 \newcommand*\led@err@EdtextWithoutFootnote}{%
250 \reledmac@error{edtext without Xfootnote. Check syntaxis.}{\@ehc}%
251 }%
252 %

err@FootnoteWithoutEdtext53 \newcommand*\led@err@FootnoteWithoutEdtext}{%
254 \reledmac@error{Xfootnote without edtext. Check syntax.}{\@ehc}%
255 }%
256 %

error@ImakeidxAfterEledmac57 \newcommand*\led@error@ImakeidxAfterEledmac}{%
258 \reledmac@error{Imakeidx must be loaded before reledmac.}{\@ehc}%
259 }%
260 %

or@IndextoolsAfterEledmac61 \newcommand*\led@error@IndextoolsAfterEledmac}{%
262 \reledmac@error{Indextools must be loaded before reledmac.}{\@ehc}%
263 }%
264 %

error@fail@patch@@makecol65 \newcommand*\led@error@fail@patch@@makecol}{%
266 \reledmac@error{Fail to patch \string\@makecol\space command.}{\@ehc}%
267 }%
268 %

ror@fail@patch@@reinserts69 \newcommand*\led@error@fail@patch@@reinserts}{%
270 \reledmac@error{Fail to patch \string\@reinserts\space command.}{\@ehc}%
271 }%
272 %

```

```

\led@error@fail@patch@@doclearpage73 \newcommand{\led@error@fail@patch@@doclearpage}{%
274 \reledmac@error{Fail to patch \string\@doclearpage\space command.}\@ehc}
%
275 }%
276 %

```

```

\led@error@fail@patch@@iiiminipage77 \newcommand{\led@error@fail@patch@@iiiminipage}{%
278 \reledmac@error{Fail to patch \string\@iiiminipage\space command.}\@ehc}
%
279 }%
280 %

```

```

\led@error@fail@patch@endminipage81 \newcommand{\led@error@fail@patch@endminipage}{%
282 \reledmac@error{Fail to patch \string@endminipage\space command.}\@ehc}%
283 }%
284 %

```

## II.8 Gobbling

Here, we define some commands which gobble their arguments.

```

\@gobblethree85 \providecommand*\@gobblethree}[3]{}
\@gobblefour86 \providecommand*\@gobblefour}[4]{}
\@gobblefive87 \providecommand*\@gobblefive}[5]{}
288 %

```

## II.9 Miscellaneous commands

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

```

289 \ifledfinal
290 \newcommand*\showlemma[1]{#1}
291 \else
292 \newcommand*\showlemma[1]{\underline{#1}}
293 \fi
294
295 %

```

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

```

296 \let\linenumberlist=\empty
297
298 %

```



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

```
299 \newcount\@l@tempcnta \newcount\@l@tempcntb
300 %
```

## 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`

```
301 \newif\ifl@dpairing
302 \newif\ifl@dpaging%
303 \newif\ifl@dprintingpages%
304 \newif\ifl@dprintingcolumns%
305 \newif\ifpst@rtedL
306 \newcount\l@dnumpsstartsL
307 %
```

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

```
308 \newif\ifledRcol
309 \newif\ifledRcol@
310 %
```

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

```
311 \newif\ifnumberingR
312 %
```

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

```
313 \newif\ifXnote@%
314 %
```

## 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.  
`\ifindtl@notenumber`

```
315 \providebool{indtl@innote}%
316 \providebool{indtl@notenumber}%
317 %
```

### 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 `\resumnumbering` 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.

```

318 \newcount\section@num
319 \section@num=0
320 \let\extensionchars=\empty
321 %

```

`\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.

```

322 \newif\ifnumbering
323 %

```

`\beginnumbering` `\initnumbering@reg` `\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 `\ifpst@rtedL` to FALSE.

```

324 \newcommand*{\beginnumbering}{%
325   \ifnumbering
326     \led@err@NumberingStarted
327   \endnumbering
328   \fi
329   \global\numberingtrue
330   \global\advance\section@num \@ne
331   \initnumbering@reg
332   \message{Section \the\section@num }%
333   \line@list@stuff{\jobname.\extensionchars\the\section@num}%
334   \l@dend@stuff
335   \setcounter{pstart}{1}
336   \ifl@dpairing
337     \global\l@dnumpstartsL \z@
338     \global\pst@rtedLfalse
339   %

```

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.

```

340   \else
341     \begingroup
342     \global\@afterindenttrue%In order to reestablish normal feature if the \
343     \initnumbering@quote
344     \ifwidthliketwocolumns%
345       \csuse{setwidthliketwocolumns@\columns@position}%
346       \csuse{setpositionliketwocolumns@\columns@position}%
347     \fi%
348     \fi
349     \gdef\eled@sections@{ }%
350     \if@noeled@sec\else%
351       \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@num}{ }{\
352       \immediate\openout\eled@sectioning@out=\jobname.eledsec\the\section@num
353       \relax%
354       \fi%
355     }
356   \newcommand*{\initnumbering@reg}{%
357     \global\pst@rtedLfalse
358     \global\l@dnumpstartsL \z@
359     \global\absline@num \z@
360     \gdef\normal@page@break{}
361     \gdef\l@prev@pb{}
362     \gdef\l@prev@nopb{}
363     \global\line@num \z@

```

```

363 \global\subline@num \z@
364 \global\@lock \z@
365 \global\sub@lock \z@
366 \global\sublines@false
367 \global\let\next@page@num=\relax
368 \global\let\sub@change=\relax
369 \resetprevline@
370 \resetprevpage@num
371 }
372
373 %

```

**\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.

```

374 \def\endnumbering{%
375   \ifnumbering
376     \global\numberingfalse
377     \normal@pars
378     \ifnum\l@dnumstartsl=0%
379       \led@err@NumberingWithoutPstart%
380     \fi%
381     \ifl@dpairing
382       \global\pst@rtedLfalse
383     \else
384       \ifx\insertlines@list\empty\else
385         \global\noteschanged@true
386       \fi
387       \ifx\line@list\empty\else
388         \global\noteschanged@true
389       \fi
390     \fi
391     \ifnoteschanged@
392       \led@mess@NotesChanged
393     \fi
394   \else
395     \led@err@NumberingNotStarted
396   \fi
397   \autoparfalse
398   \if@noeled@sec\else%
399     \immediate\closeout\eled@sectioning@out%
400   \fi%
401   \ifl@dpairing\else
402     \global\l@dnumstartsl=\z@%
403   \endgroup
404   \fi
405 }
406 %

```

`\pausenumbering` The `\pausenumbering` macro is just the same as `\endnumbering`, but with the `\ifnumbering` flag set to true, to show that numbering continues across the gap.<sup>24</sup>

`\resumenumbering`

```
407 \newcommand{\pausenumbering}{%
408   \ifautopar\global\autopar@pausetrue\fi%
409   \endnumbering\global\numberingtrue}
410 %
```

The `\resumenumbering` 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 `\resumenumbering` to ensure that `\pausenumbering` was actually invoked.

```
411 \newcommand*{\resumenumbering}{%
412   \ifnumbering
413     \ifautopar@pause\autopar\fi
414     \global\pst@rtedLtrue
415     \global\advance\section@num \@ne
416     \led@mess@SectionContinued{\the\section@num}%
417     \line@list@stuff{\jobname.\extensionchars\the\section@num}%
418     \l@dend@stuff
419     \ifl@dpairing\else%
420       \begingroup%
421       \initnumbering@quote%
422       \ifwidthliketwocolumns%
423         \csuse{setwidthliketwocolumns@\columns@position}%
424         \csuse{setpositionliketwocolumns@\columns@position}%
425       \fi%
426     \fi%
427   \else
428     \led@err@NumberingShouldHaveStarted
429     \endnumbering
430     \beginnumbering
431   \fi}
432
433
434 %
```

## 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.

<sup>24</sup>Peter Wilson’s thanks to Wayne Sullivan, who suggested the idea behind these macros.

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 `LATEX3` 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.

```
435 \newcommand*{\list@create}[1]{%
436   \global\let#1=\empty%
437 }%
438 %
```

**`\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 .

```
439 \newcommand*{\list@clear}[1]{%
440   \global\let#1=\empty%
441 }
442 %
```

**`\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  
**`\led@toksb`** of a counter. `\xright@appenditem` creates global control sequences, like `\xdef`, and uses two temporary token-list registers, `\@toksa` and `\@toksb`.

```
443 \newtoks\led@toksa \newtoks\led@toksb
444 \global\led@toksa={\}
445 \long\def\xright@appenditem#1\to#2{%
446   \global\led@toksb=\expandafter{#2}%
447   \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
448   \global\led@toksb={}}
449 %
```

**`\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`.

```
450 \long\def\xleft@appenditem#1\to#2{%
451   \global\led@toksb=\expandafter{#2}%
452   \xdef#2{\the\led@toksa\expandafter{#1}\the\led@toksb}%
453   \global\led@toksb={}}
454 %
```

**`\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.

```

455 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
456 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
457
458 %

```

## 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:

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

`\ifbypage@`  
`\bypage@true`  
`\bypage@false` reledmac will use the line-of-section system unless instructed otherwise.

```

459 \newif\ifbypage@
460 \newif\ifbypstart@
461 %

```

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@R462 \newif\ifbypage@R
\ifbypstart@R463 \newif\ifbypstart@R
464 %

```

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

```

465 \newcommand*{\lineation}[1]{
466 %

```

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

```

467 \ifnumbering
468 \led@err@LineationInNumbered
469 \else
470 %

```

If the argument is page.

```

471 \def\@tempa{#1}\def\@tempb{page}%
472 \ifx\@tempa\@tempb
473   \global\bypage@true
474   \global\bystart@false
475   \unless\ifnocritical@%
476     \Xpstart[] [false]%
477   \fi%
478 %

```

If the argument is pstart.

```

479 \else
480   \def\@tempb{pstart}%
481   \ifx\@tempa\@tempb
482     \global\bypage@false
483     \global\bystart@true
484     \unless\ifnocritical@%
485       \Xpstart%
486     \fi%
487 %

```

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

```

488 \else
489   \def\@tempb{section}
490   \ifx\@tempa\@tempb
491     \global\bypage@false
492     \global\bystart@false
493     \unless\ifnocritical@%
494       \Xpstart[] [false]%
495     \fi%
496 %

```

In other case, it is an error.

```

497 \else
498   \led@warn@BadLineation
499   \fi
500 \fi
501 \fi
502 \fi}}
503 %

```

## 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@getline@margin`

```

504 \newcount\line@margin
505

```



```

506 \newcommand*{\linenummargin}[1]{%
507   \l@dgetline@margin{#1}%
508   \ifnum\l@dtempcntb>\m@ne
509     \ifledRcol
510       \global\line@marginR=\l@dtempcntb
511       \led@warn@setting@in@rightside{\linenummargin}%
512     \else
513       \global\line@margin=\l@dtempcntb
514     \fi
515   \fi}}
516
517 \newcommand*{\l@dgetline@margin}[1]{%
518   \def\@tempa{#1}\def\@tempb{left}%
519   \ifx\@tempa\@tempb
520     \l@dtempcntb \z@
521   \else
522     \def\@tempb{right}%
523     \ifx\@tempa\@tempb
524       \l@dtempcntb \@ne
525     \else
526       \def\@tempb{outer}%
527       \ifx\@tempa\@tempb
528         \l@dtempcntb \tw@
529       \else
530         \def\@tempb{inner}%
531         \ifx\@tempa\@tempb
532           \l@dtempcntb \thr@@
533         \else
534           \led@warn@BadLinenummargin
535           \l@dtempcntb \m@ne
536         \fi
537       \fi
538     \fi
539   \fi}
540
541 %

```

### 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.

```

542 \newcounter{firstlinenum}
543   \setcounter{firstlinenum}{5}
544 \newcounter{linenumincrement}
545   \setcounter{linenumincrement}{5}

```

546 %

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

```
547 \newcounter{firstsublinenum}
548   \setcounter{firstsublinenum}{5}
549 \newcounter{sublinenumincrement}
550   \setcounter{sublinenumincrement}{5}
551
552 %
```

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

```
\linenumincrement
\firstsublinenum
\sublinenumincrement
553 \newcommand*\firstlinenum[1]{%
554   \ifledRcol%
555     \setcounter{firstlinenumR}{#1}%
556     \led@warn@setting@in@rightside{\firstlinenum}
557   \else%
558     \setcounter{firstlinenum}{#1}%
559   \fi%
560 }
561
562 \newcommand*\linenumincrement[1]{%
563   \ifledRcol%
564     \setcounter{linenumincrementR}{#1}%
565     \led@warn@setting@in@rightside{\linenumincrement}
566   \else%
567     \setcounter{linenumincrement}{#1}%
568   \fi%
569 }
570 \newcommand*\firstsublinenum[1]{%
571   \ifledRcol%
572     \setcounter{firstsublinenumR}{#1}%
573     \led@warn@setting@in@rightside{\firstsublinenum}
574   \else%
575     \setcounter{firstsublinenum}{#1}%
576   \fi%
577 }
578 \newcommand*\sublinenumincrement[1]{%
579   \ifledRcol%
580     \setcounter{sublinenumincrementR}{#1}%
581     \led@warn@setting@in@rightside{\sublinenumincrement}
582   \else%
583     \setcounter{sublinenumincrement}{#1}%
584   \fi%
585 }
586
587 %
```

## 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.

```

588 \newcount\lock@disp
589 \newcommand{\lockdisp}[1]{\%
590   \l@dgetlock@disp{#1}%
591   \ifnum\@l@dttempcntb>\m@ne
592     \global\lock@disp=\@l@dttempcntb
593   \else
594     \led@warn@BadLockdisp
595   \fi}}
596 \newcommand*{\l@dgetlock@disp}[1]{
597   \def\@tempa{#1}\def\@tempb{first}%
598   \ifx\@tempa\@tempb
599     \@l@dttempcntb \z@
600   \else
601     \def\@tempb{last}%
602     \ifx\@tempa\@tempb
603       \@l@dttempcntb \@ne
604     \else
605       \def\@tempb{all}%
606       \ifx\@tempa\@tempb
607         \@l@dttempcntb \tw@
608       \else
609         \@l@dttempcntb \m@ne
610       \fi
611     \fi
612   \fi}
613
614 %

```

`\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.

```

615 \newcount\sublock@disp
616 \newcommand{\sublockdisp}[1]{\%
617   \l@dgetlock@disp{#1}%
618   \ifnum\@l@dttempcntb>\m@ne
619     \global\sublock@disp=\@l@dttempcntb
620   \else
621     \led@warn@BadSublockdisp
622   \fi}}
623
624 %

```

## 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.

`\sublinenumrep`

`\sublinenumr@p`

```

625 \newcommand*{\linenumberstyle}[1]{%
626   \def\linenumrep##1{\@nameuse{##1}}
627 \newcommand*{\sublinenumberstyle}[1]{%
628   \def\sublinenumrep##1{\@nameuse{##1}}
629 %

```

Initialise the number styles to arabic.

```

630 \linenumberstyle{arabic}
631 \let\linenumr@p\linenumrep
632 \sublinenumberstyle{arabic}
633 \let\sublinenumr@p\sublinenumrep
634
635 %

```

## V.6 Line number printing

`\leftlinenum` `\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.

`\rightlinenum`

`\linenumsep`

`\numlabfont`

`\ledlinenum`

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 subline) number.

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

```

636 \newlength{\linenumsep}
637 \setlength{\linenumsep}{1pc}
638 \newcommand*{\numlabfont}{\normalfont\scriptsize}
639 \newcommand*{\ledlinenum}{%
640   \bgroup%
641   \ifluatex%
642     \texdir TLT%

```

```

643 \fi%
644 \numlabfont\linenumrep{\line@num}%
645 \ifsublines@
646 \ifnum\subline@num>0\relax
647 \unskip%
648 \Xsublinesep@side%
649 \sublinenumrep{\subline@num}%
650 \fi
651 \fi%
652 \egroup%
653 }%
654
655 \newcommand*{\leftlinenum}{%
656 \ledlinenum
657 \kern\linenumsep}
658 \newcommand*{\rightlinenum}{%
659 \kern\linenumsep
660 \ledlinenum}
661
662 %

```

## 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  $\TeX$  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`.

```

663 \newcount\line@num
664 %

```

**\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.

```

665 \newcount\subline@num
666 %

```

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

`\sublines@true` 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.

```

667 \newif\ifsublines@
668 %

```

`\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.

```

669 \newcount\absline@num
670 %

```

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.

```

671 \newcount\@lock
672 \newcount\sub@lock
673 %

```

## 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|0T1/cmr/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:

```

674 \list@create{\line@list}
675 \list@create{\insertlines@list}
676 \list@create{\actionlines@list}
677 \list@create{\actions@list}
678
679 %

```

<pre> \page@num \endpage@num \endline@num \endsubline@num </pre>	<p>We will need some counts while we read the line-list, for the page number and the ending page, line, and sub-line numbers. Some of these will be used again later on, when we are acting on the data in our list macros.</p>
<pre> 680 \newcount\page@num 681 \newcount\endpage@num 682 \newcount\endline@num </pre>	



```

683 \newcount\endsubline@num
684 %

```

`\ifnoteschanged@` If the number of the footnotes in a section is different from what it was during the last run, or if this is the very first time you've run  $\LaTeX$ , on this file, the information from 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  $\LaTeX$  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.

```

685 \newif\ifnoteschanged@
686 %

```

`\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@87 \newcommand*{\resetprevline@}{%
688     \def\do##1{\global\csundef{prevline##1}}%
689     \dolistloop{\@series}%
690 }
691 %

```

`\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@92 \newcommand*{\resetprevpage@num}{%
693     \def\do##1{\ifcsdef{prevpage##1@num}{\global\csname prevpage##1@num\
endcsname=0}{}}%
694     \dolistloop{\@series}%
695 }
696 %

```

## 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.

```

697 \newread\@inputcheck
698 \newcommand*{\read@linelist}[1]{%
699   \ifledRcol%
700   \list@clearing@regR%
701   \else%
702   \list@clearing@reg%
703   \fi%
704 %

```

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).

```

705   \list@clear{\maxlinesinpar@list}
706 %

```

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  $\text{\TeX}$  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  $\text{\LaTeX}$  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 `\resumenumbering`, 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.

```

707 \get@linelistfile{#1}%
708 \endgroup
709 %

```

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.

```

710 \ifledRcol
711   \global\page@numR=\m@ne
712   \ifx\actionlines@listR\empty
713     \gdef\next@actionlineR{1000000}%
714   \else
715     \glp\actionlines@listR\to\next@actionlineR
716     \glp\actions@listR\to\next@actionR
717   \fi
718 \else
719   \global\page@num=\m@ne
720   \ifx\actionlines@list\empty

```

```

721 \gdef\next@actionline{1000000}%
722 \else
723 \glp\actionlines@list\to\next@actionline
724 \glp\actions@list\to\next@action
725 \fi
726 \fi
727 }
728 %

```

`\list@clearing@reg` Clears the lists for `\read@linelist`

```

729 \newcommand*\list@clearing@reg{%
730 \list@clear{\line@list}%
731 \list@clear{\insertlines@list}%
732 \list@clear{\actionlines@list}%
733 \list@clear{\actions@list}%
734 \list@clear{\linesinpar@listL}%
735 \list@clear{\linesonpage@listL}%
736 }%
737 %

```

`\get@linelistfile` reledmac can take advantage of the  $\TeX$  ‘safe file input’ macros to get the line-list file.

```

738 \newcommand*\get@linelistfile}[1]{%
739 \InputIfFileExists{#1}{%
740 \global\noteschanged@false
741 \begingroup
742 \catcode`\[=1 \catcode`\]=2
743 \makeatletter \catcode`\^M=9}{%
744 \led@warn@NoLineFile{#1}%
745 \global\noteschanged@true
746 \begingroup}%
747 }
748 %
749 %

```

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  $\TeX$  for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the `\pausenumbers` and `\resumenumbers` macros to help you if you run into macro memory limitations (see 4.2.7 p. 17 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 refer 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.

```

750 \newcommand{\line@list@version}[1]{%
751   \ifStrEq{#1}{\this@line@list@version}%
752   {}%
753   {\ifledRcol%
754     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
755     \else%
756     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
757     \fi%
758     \endinput%
759   }%
760 }%
761 %

```

`\@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`.

```

763 \newcommand*{\@nl}[2]{%
764   \fix@page{#1}%
765   \ifledRcol%
766     \@nl@regR%
767   \else%
768     \@nl@reg%
769   \fi%
770 }
771 \newcommand*{\@nl@reg}{%
772   \ifx\l@dchset@num\relax \else
773     \advance\absline@num \@ne
774     \set@line@action
775     \let\l@dchset@num=\relax
776     \advance\absline@num \m@ne
777     \advance\line@num \m@ne
778   \fi
779   %

```

First increment the absolute line-number, and perform deferred actions relating to page starts and sub-lines.

```

780   \advance\absline@num \@ne
781     \ifx\next@page@num\relax \else
782       \page@action
783       \let\next@page@num=\relax
784     \fi
785     \ifx\sub@change\relax \else
786       \ifnum\sub@change>\z@
787         \sublines@true
788       \else
789         \sublines@false
790       \fi
791       \sub@action
792       \let\sub@change=\relax
793     \fi
794   %

```

Fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

795   \ifcase\@lock
796     \or
797       \@lock \tw@
798     \or \or
799       \@lock \z@
800     \fi
801   \ifcase\sub@lock
802     \or
803       \sub@lock \tw@
804     \or \or
805       \sub@lock \z@

```

```

806         \fi
807 %
      Now advance the visible line number, unless it has been locked.
808         \ifsublines@
809             \ifnum\sub@lock<\tw@
810                 \advance\subline@num \@ne
811             \fi
812         \else
813             \ifnum\@lock<\tw@
814                 \advance\line@num \@ne \subline@num \z@
815             \fi
816         \fi}
817 %
818 %

```

`\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`.

```

819 \newcount\last@page@num
820 \last@page@num=-10000
821
822 \newcommand*{\fix@page}[1]{%
823     \iflabeledRcol
824         \ifnum #1=\last@page@numR
825         \else
826             \ifbypage@R
827                 \line@numR \z@ \subline@numR \z@
828             \fi
829             \page@numR=#1\relax
830             \last@page@numR=#1\relax
831             \def\next@page@numR{#1}%
832         \fi
833     \else
834         \ifnum #1=\last@page@num
835         \else
836             \ifbypage@
837                 \line@num \z@ \subline@num \z@
838             \fi
839             \page@num=#1\relax
840             \last@page@num=#1\relax
841             \def\next@page@num{#1}%
842             \listxadd{\normal@page@break}{\the\absline@num}
843         \fi
844     \fi}
845 %

```

`\@pend` These do not do anything at this point, but will have been added to the auxiliary file(s)  
`\@pendR` if the `reledpar` package has been used. They are just here to stop `reledmac` from  
`\@lopL` moaning if the `reledpar` is used for one run and then not for the following one.  
`\@lopR`

```

846 \newcommand*{\@pend}[1]{}
847 \newcommand*{\@pendR}[1]{}
848 \newcommand*{\@lopL}[1]{}
849 \newcommand*{\@lopR}[1]{}
850
851 %

```

**\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 `\@nl` of the necessary action.

**\sub@off**

```

852 \newcommand*{\sub@on}{\ifsublines@
853   \let\sub@change=\relax
854   \else
855     \def\sub@change{1}%
856   \fi}
857 \newcommand*{\sub@off}{\ifsublines@
858   \def\sub@change{-1}%
859   \else
860     \let\sub@change=\relax
861   \fi}
862
863 %

```

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

```

864
865 \newcommand*{\@adv}[1]{%
866   \ifsublines@
867     \ifledRcol
868       \advance\subline@numR by #1\relax
869       \ifnum\subline@numR<\z@
870         \led@warn@BadAdvancelineSubline
871         \subline@numR \z@
872       \fi
873     \else
874       \advance\subline@num by #1\relax
875       \ifnum\subline@num<\z@
876         \led@warn@BadAdvancelineSubline
877         \subline@num \z@
878       \fi
879     \fi
880   \else
881     \ifledRcol
882       \advance\line@numR by #1\relax
883       \ifnum\line@numR<\z@
884         \led@warn@BadAdvancelineLine
885         \line@numR \z@
886       \fi

```

```

887 \else
888   \advance\line@num by #1\relax
889   \ifnum\line@num<\z@
890     \led@warn@BadAdvancelineLine
891     \line@num \z@
892   \fi
893 \fi
894 \fi
895 \set@line@action}
896
897 %

```

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

```

898
899 \newcommand*\@set}[1]{%
900   \ifledRcol
901     \ifsublines@
902       \subline@numR=#1\relax
903     \else
904       \line@numR=#1\relax
905     \fi
906     \set@line@action
907   \else
908     \ifsublines@
909       \subline@num=#1\relax
910     \else
911       \line@num=#1\relax
912     \fi
913     \set@line@action
914   \fi}
915
916 %

```

**\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.

```

917
918 \newcommand*\l@d@set}[1]{%
919   \ifledRcol
920     \line@numR=#1\relax
921     \advance\line@numR \@ne
922     \def\l@dchset@num{#1}
923   \else
924     \line@num=#1\relax
925     \advance\line@num \@ne
926     \def\l@dchset@num{#1}

```



```

927 \fi}
928 \let\l@dchset@num\relax
929
930 %

```

**\page@action** \page@action adds an entry to the action-code list to change the page number.

```

931
932 \newcommand*{\page@action}{%
933 \ifledRcol
934 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
935 \xright@appenditem{\next@page@numR}\to\actions@listR
936 \else
937 \xright@appenditem{\the\absline@num}\to\actionlines@list
938 \xright@appenditem{\next@page@num}\to\actions@list
939 \fi}
940 %

```

**\set@line@action** \set@line@action adds an entry to the action-code list to change the visible line number.

```

941
942 \newcommand*{\set@line@action}{%
943 \ifledRcol
944 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
945 \ifsublines@
946 \l@dtempcnta=-\subline@numR
947 \else
948 \l@dtempcnta=-\line@numR
949 \fi
950 \advance\l@dtempcnta by -5000\relax
951 \xright@appenditem{\the\l@dtempcnta}\to\actions@listR
952 \else
953 \xright@appenditem{\the\absline@num}\to\actionlines@list
954 \ifsublines@
955 \l@dtempcnta=-\subline@num
956 \else
957 \l@dtempcnta=-\line@num
958 \fi
959 \advance\l@dtempcnta by -5000\relax
960 \xright@appenditem{\the\l@dtempcnta}\to\actions@list
961 \fi}
962 %

```

**\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.

```

963
964 \newcommand*{\sub@action}{%
965 \ifledRcol

```

```

966 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
967 \ifsublines@
968 \xright@appenditem{-1001}\to\actions@listR
969 \else
970 \xright@appenditem{-1002}\to\actions@listR
971 \fi
972 \else
973 \xright@appenditem{\the\absline@num}\to\actionlines@list
974 \ifsublines@
975 \xright@appenditem{-1001}\to\actions@list
976 \else
977 \xright@appenditem{-1002}\to\actions@list
978 \fi
979 \fi}
980 %

```

`\lock@on` `\lock@on` adds an entry to the action-code list to turn line number locking on. The `\do@lockon` current setting of the sub-lineation flag tells us whether this applies to line numbers or `\do@lockonL` 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.

```

981 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
982
983 \newcommand*{\do@lockon}{%
984 \ifx\next\lock@off
985 \global\let\lock@off=\skip@lockoff
986 \else
987 \ifledRcol
988 \do@lockonR
989 \else
990 \do@lockonL
991 \fi
992 \fi}
993
994
995 \newcommand*{\do@lockonL}{%
996 \xright@appenditem{\the\absline@num}\to\actionlines@list
997 \ifsublines@
998 \xright@appenditem{-1005}\to\actions@list
999 \ifnum\sub@lock=\z@
1000 \sub@lock \@ne
1001 \else
1002 \ifnum\sub@lock=\thr@@
1003 \sub@lock \@ne
1004 \fi
1005 \fi}

```

```

1006 \else
1007 \xright@appenditem{-1003}\to\actions@list
1008 \ifnum\@lock=\z@
1009 \@lock \@ne
1010 \else
1011 \ifnum\@lock=\thr@@
1012 \@lock \@ne
1013 \fi
1014 \fi
1015 \fi}
1016
1017 %

```

**\lock@off** \lock@off adds an entry to the action-code list to turn line number locking off.

```

\do@lockoff
\do@lockoffL
\skip@lockoff
1018 \newcommand*{\do@lockoffL}{%
1019 \xright@appenditem{\the\absline@num}\to\actionlines@list
1020 \ifsublines@
1021 \xright@appenditem{-1006}\to\actions@list
1022 \ifnum\sub@lock=\tw@
1023 \sub@lock \thr@@
1024 \else
1025 \sub@lock \z@
1026 \fi
1027 \else
1028 \xright@appenditem{-1004}\to\actions@list
1029 \ifnum\@lock=\tw@
1030 \@lock \thr@@
1031 \else
1032 \@lock \z@
1033 \fi
1034 \fi}
1035
1036 \newcommand*{\do@lockoff}{%
1037 \ifledRcol
1038 \do@lockoffR
1039 \else
1040 \do@lockoffL
1041 \fi}
1042 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}
1043 \global\let\lock@off=\do@lockoff
1044
1045 %

```

**\n@num** These macros implement the \skipnumbering command. They use action code 1007.

```

1046 \newcommand*{\n@num}{%
1047 \ifledRcol%
1048 \xright@appenditem{\the\absline@numR}\to\actionlines@listR

```

```

1049 \xright@appenditem{-1007}\to\actions@listR
1050 \else%
1051 \xright@appenditem{\the\absline@num}\to\actionlines@list%
1052 \xright@appenditem{-1007}\to\actions@list%
1053 \fi%
1054 }%
1055
1056 %

```

**\n@num@stanza** This macro implements the `\skipnumbering` for stanza command. It uses action code 1008.

```

1057 \newcommand*{\n@num@stanza}{%
1058 \ifledRcol%
1059 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1060 \xright@appenditem{-1008}\to\actions@listR%
1061 \else%
1062 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1063 \xright@appenditem{-1008}\to\actions@list%
1064 \fi%
1065 }
1066 %

```

**\ifl@dhidenumber** `\hidenum` hides number in margin. It uses action code 1009.

**\hidenum**

```

1067 \newif\ifl@dhidenumber
1068 \newcommand*{\hidenum}{%
1069 \ifledRcol%
1070 \write\linenum@outR{\string\hide@num}%
1071 \else%
1072 \write\linenum@out{\string\hide@num}%
1073 \fi%
1074 }%
1075 \newcommand*{\hide@num}{%
1076 \ifledRcol%
1077 \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1078 \xright@appenditem{-1009}\to\actions@listR%
1079 \else%
1080 \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1081 \xright@appenditem{-1009}\to\actions@list%
1082 \fi%
1083 }
1084 %

```

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

- **#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`.

```

1085 \newcount\insert@count
1086 %

```

- #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.

```

1087 \newcommand*\@dummy@ref}[2]{#2}
1088 %

```

**\@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.

```

1089 \newcommand*\@ref}[2]{%
1090   \ifledRcol%
1091     \@ref@regR{#1}{#2}%
1092   \else%
1093     \@ref@reg{#1}{#2}%
1094   \fi%
1095 }%
1096 \newcommand*\@ref@reg}[2]{%
1097   \global\insert@count=#1\relax
1098   \global\advance\@edtext@level by 1%
1099   \loop\ifnum\insert@count>\z@
1100     \xright@appenditem{\the\absline@num}\to\insertlines@list
1101     \global\advance\insert@count \m@ne
1102   \repeat
1103 %

```

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.

```

1104 \begingroup
1105   \let\@ref=\dummy@ref
1106   \let\@lopL\@gobble
1107   \let\page@action=\relax
1108   \let\sub@action=\relax
1109   \let\set@line@action=\relax
1110   \let\@lab=\relax
1111   \let\@lemma=\relax%
1112   \let\@sw\@gobblethree%
1113   #2
1114   \global\endpage@num=\page@num
1115   \global\endline@num=\line@num
1116   \global\endsubline@num=\subline@num

```

```

1117 \endgroup
1118 %

```

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

```

1119 \xright@appenditem%
1120   {\the\page@num|\the\line@num|%
1121    \ifsublines@ \the\subline@num \else 0\fi}%
1122   \the\endpage@num|\the\endline@num|%
1123   \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1124 %

```

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.

```

1125 \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
1126   @edtext@level\endcsname}%
1127 \providebool{lemmacommand@\the\@edtext@level}%
1128 \boolfalse{lemmacommand@\the\@edtext@level}%
1129 %

```

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

```

1129 #2%
1130 %

```

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.

```

1131 \ifnum\@edtext@level>0%
1132   \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
1133     csname sw@list@edtext@\the\@edtext@level\endcsname}}%
1134   \ifcsundef{sw@list@edtext@\the\@edtext@level}{\create@this@edtext@level}%
1135   \letcs{\@tmp}{sw@list@edtext@\the\@edtext@level}%
1136   \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}%
1137   \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
1138   \global\cslet{sw@list@edtext@\the\@edtext@level}{\@tmp}%
1139   \fi%
1140 %

```

Decrease edtext level counter.

```

1140 \global\advance\@edtext@level by -1%
1141 %
1142 }
1143
1144 %

```

## 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`.

```
1145 \newwrite\linenum@out
1146 %
```

`\iffirst@linenum@out@`  
`\first@linenum@out@true`  
`\first@linenum@out@false`

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.

```
1147 \newif\iffirst@linenum@out@
1148 \first@linenum@out@true
1149 %
```

`\this@line@list@version` The commands allowed in the line-list file and their arguments can change between two version 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 a older version, that means the commands used inside are deprecated, and we can't use them.

```
1150 \newcommand{\this@line@list@version}{5}%
1151 %
```

`\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.

```
1152 \newcommand*{\line@list@stuff}[1]{%
1153 %
```

First, use the commands of the previous section to interpret the line-list file from the last run.

```
1154 \read@linelist{#1}%
1155 %
```

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.

```

1156 \iffirst@linenum@out@
1157 \immediate\closeout\linenum@out%
1158 \global\first@linenum@out@false%
1159 \immediate\openout\linenum@out=#1\relax%
1160 \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}}%
1161 \ifl@dpaging%
1162 \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}}%
1163 \fi%
1164 \else
1165 %

```

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.

```

1166 \if@minipage%
1167 \leavevmode%
1168 \fi%
1169 \closeout\linenum@out%
1170 \openout\linenum@out=#1\relax%
1171 \fi}
1172
1173 %

```

`\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.

```

1174 \newcommand*{\new@line}{%
1175 \IfStrEq{\led@pb@setting}{after}%
1176 {\xifinlist{\the\absline@num}{\l@prev@nopb}%
1177 {\xifinlist{\the\absline@num}{\normal@page@break}%
1178 {\numdef{\@next@page}{\c@page+1}%
1179 \write\linenum@out{\string\@nl[\@next@page][\@next@page}}}%
1180 }%
1181 {\write\linenum@out{\string\@nl[\the\c@page][\thepage}}}%
1182 }%
1183 {\write\linenum@out{\string\@nl[\the\c@page][\thepage}}}%
1184 }%
1185 \IfStrEq{\led@pb@setting}{before}%
1186 {\numdef{\next@absline}{\the\absline@num+1}%
1187 \xifinlist{\next@absline}{\l@prev@nopb}%
1188 {\xifinlist{\the\absline@num}{\normal@page@break}%
1189 {\numdef{\nc@page}{\c@page+1}%
1190 \write\linenum@out{\string\@nl[\nc@page][\nc@page}}}%
1191 }%
1192 {\write\linenum@out{\string\@nl[\the\c@page][\thepage}}}%

```



```

1193 }%
1194 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1195 }%
1196 {}%
1197 \IfStrEqCase{\led@pb@setting}{\before}{\relax}{\after}{\relax}}{\write\
linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1198 }
1199
1200 %

```

**\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  
**\flag@end** 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.

```

1201 \newif\if@noneed@Footnote%
1202
1203 \newcommand*{\flag@start}{%
1204   \ifledRcol%
1205     \edef\next{\write\linenum@outR{%
1206       \string\@ref[\the\insert@countR] []}%
1207     \next%
1208     \ifnum\insert@countR<1%
1209       \if@noneed@Footnote\else%
1210         \led@err@EdtextWithoutFootnote%
1211       \fi%
1212     \fi%
1213   \else%
1214     \edef\next{\write\linenum@out{%
1215       \string\@ref[\the\insert@count] []}%
1216     \next%
1217     \ifnum\insert@count<1%
1218       \if@noneed@Footnote\else%
1219         \led@err@EdtextWithoutFootnote%
1220       \fi%
1221     \fi%
1222   \fi}%
1223
1224 %

```

**\startsub** \startsub and \endsub turn sub-lineation on and off, by writing appropriate instruc-  
**\endsub** tions 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.

```

1225
1226
1227 \newcommand*{\startsub}{\dimen0\lastskip
1228   \ifdim\dimen0>Opt \unskip \fi
1229   \ifledRcol \write\linenum@outR{\string\sub@on}%
1230   \else      \write\linenum@out{\string\sub@on}%
1231   \fi
1232   \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1233 \def\endsub{\dimen0\lastskip
1234   \ifdim\dimen0>Opt \unskip \fi
1235   \ifledRcol \write\linenum@outR{\string\sub@off}%
1236   \else      \write\linenum@out{\string\sub@off}%
1237   \fi
1238   \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1239
1240 %

```

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

```

1241 \newcommand*{\advanceline}[1]{\leavevmode%
1242   \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
1243   \else      \write\linenum@out{\string\@adv[#1]}%
1244   \fi}%
1245 }
1246 %

```

**\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.

```

1247
1248 \newcommand*{\setline}[1]{%
1249   \leavevmode%
1250   \ifnum#1<\z@
1251     \led@warn@BadSetline
1252   \else
1253     \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1254     \else      \write\linenum@out{\string\@set[#1]}%
1255     \fi
1256   \fi}
1257
1258 %

```

**\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.

```

1259
1260 \newcommand*{\setlinenum}[1]{%
1261   \ifnum#1<\z@
1262     \led@warn@BadSetlinenum
1263   \else
1264     \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1265     \else      \write\linenum@out{\string\l@d@set[#1]} \fi
1266   \fi}
1267
1268 %

```

**\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.

**\endlock**

```

1269
1270 \newcommand*{\startlock}{%
1271   \ifledRcol \write\linenum@outR{\string\lock@on}%
1272   \else      \write\linenum@out{\string\lock@on}%
1273   \fi}
1274 \def\endlock{%
1275   \ifledRcol \write\linenum@outR{\string\lock@off}%
1276   \else      \write\linenum@out{\string\lock@off}%
1277   \fi}
1278 %

```

**\ifl@dskipnumber** In numbered text `\skipnumbering` will suspend the numbering for that particular line.

**\ifl@dskipversenumber**

```

1279 \l@dskipnumbertrue
1280 \l@dskipnumberfalse
1281 \skipnumbering
1282 \newif\ifl@dskipnumber
1283 \newif\ifl@dskipversenumber%
1284 \newcommand*{\skipnumbering}{%
1285   \leavevmode%
1286   \ifledRcol%
1287     \ifinstanza%
1288       \write\linenum@outR{\string\n@num@stanza}%
1289     \else%
1290       \write\linenum@outR{\string\n@num}%
1291     \fi%
1292     \advanceline{-1}%
1293   \else%
1294     \ifinstanza%
1295       \write\linenum@out{\string\n@num@stanza}%
1296     \else%
1297       \write\linenum@out{\string\n@num}%
1298     \fi%
1299     \advanceline{-1}%

```

```

1297 \fi%
1298 }%
1299
1300 %

```

## 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. 128). 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 can not 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 accomodate 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.

```
1301 \list@create{\end@lemmas}
1302 %
```

`\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.

```
1303 \newcommand{\dummy@edtext}[2]{#1}
1304 %
```

`\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`.

```
1305 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1306 %
```

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`  
`\morenoexpands`

We need to turn off macro expansion for certain sorts of macros we are likely to see within the lemma and within the notes.

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.<sup>25</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 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 $\TeX$ .)

```

1307 \newcommand*{\no@expands}{%
1308   \let\select@lemmafont=0%
1309   \let\startsub=\relax \let\endsub=\relax
1310   \let\startlock=\relax \let\endlock=\relax
1311   \let\edlabel=\@gobble
1312   \let\setline=\@gobble \let\advanceline=\@gobble
1313   \let\sameword\sameword@inedtext%
1314   \let\edtext=\dummy@edtext

```

<sup>25</sup>Since ‘control sequences equivalent to characters are not expandable’—*The TeXbook*, answer to Exercise 20.14.

```

1315 \l@dtabnoexpands
1316 \morenoexpands}
1317 \let\morenoexpands=\relax
1318
1319 %

```

**\@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.

```

1320 \newcommand{\@tag}{}
1321 %

```

**\@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.

```

1322 \newcount\@edtext@level%
1323 \@edtext@level=0%
1324 %

```

**\theedtext** The edtext counter is increased at each \edtext command. It is used to add to insert hyperlinks between a notes and the lemma.

```

1325 \newcounter{edtext}
1326 \renewcommand{\theedtext}{edtxt@arabic{edtext}}%
1327 %

```

**\edtext** When executed, \edtext first ensures that we are in horizontal mode.

```

1328 \newcommand{\edtext}[2]{\leavevmode%
1329 %

```

Then, check if we are in a numbered paragraph (\pstart...\pend)..

```

1330 \ifnumberedpar%
1331 %

```

We increase the \@edtext@level  $\TeX$  counter to know in which level of \edtext we are.

```

1332 \global\advance\@edtext@level by 1%
1333 %

```

We also increase the edtext  $\TeX$  counter to insert hypertarget if the hyperref package is loaded.

```

1334 \stepcounter{edtext}%
1335 %

```

By default, we do not use \lemma

```

1336 \global\@lemmacommand@false%
1337 %

```

```

1338 \begingroup%
1339 %

```

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`.

```

1340 \ifledRcol%
1341 \ifcvoid{sw@list@edtextR@the\@edtext@level}%
1342 {\global\let\sw@inthisedtext\empty}%
1343 {\expandafter\gl@p\csname sw@list@edtextR@the\@edtext@level\
endcsname\to\sw@inthisedtext}%
1344 \else%
1345 \ifcvoid{sw@list@edtext@the\@edtext@level}%
1346 {\global\let\sw@inthisedtext\empty}%
1347 {\expandafter\gl@p\csname sw@list@edtext@the\@edtext@level\
endcsname\to\sw@inthisedtext}%
1348 \fi%
1349 %

```

`\@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.

```

1350 \global\renewcommand{\@tag}{%
1351 \no@expands #1%
1352 }%
1353 %

```

`\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`.

```

1354 \set@line%
1355 %

```

`\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`.

```

1356 \ifledRcol \global\insert@countR \z@%
1357 \else \global\insert@count \z@ \fi%
1358 %

```

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.



```

1359 \ignorespaces #2\relax%
1360 %

```

With polyglossia, you must track whether the language reads left to right (English) or right to left (Arabic).

```

1361 \@ifundefined{xpg@main@language}{%if not polyglossia
1362 \flag@start}%
1363 {\if@RTL\flag@end\else\flag@start\fi%
1364 }%
1365 %

```

We write in the numbered file whether the current `\edtext` has a `\lemma` in the the second argument.

```

1366 \if@lemmacommand@%
1367 \ifledRcol%
1368 \write\linenum@outR{\string\@lemma}%
1369 \else%
1370 \write\linenum@out{\string\@lemma}%
1371 \fi%
1372 \fi%
1373 %

```

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.

```

1374 \endgroup%
1375 \ifdef{\hypertarget}%
1376 {%
1377 \csedef{thisedtext@the\@edtext@level}{\theedtext}%
1378 \Hy@raisedlink{\hypertarget{\csuse{thisedtext@the\@edtext@level}:
start}}}%
1379 \showlemma{#1}%
1380 \Hy@raisedlink{\hypertarget{\csuse{thisedtext@the\@edtext@level}:
end}}}%
1381 }%
1382 {%
1383 \showlemma{#1}%
1384 }%
1385 %

```

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.

```

1386 \ifx\end@lemmas\empty \else%
1387 \glp\end@lemmas\to\x@lemma%

```

```

1388     \x@lemma%
1389     \global\let\x@lemma=\relax%
1390     \fi%
1391     \ifundefined{xpg@main@language}{%if not polyglossia
1392         \flag@end}%
1393         {\if@RTL\flag@start\else\flag@end\fi% With polyglossia, you must
1394         track whether the language reads left to right (English) or right to left
1395         (Arabic).
1396     }%
1397 %

```

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`.

```

1396     \global\@noneed@Footnotefalse%
1397     \global\advance\@edtext@level by -1%
1398     \global\@lemmacommand@false%
1399 %

```

If we are outside of a numbered paragraph, we send error message and print the first argument.

```

1400     \else%
1401     \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
1402     led@err@edtextoutsidepstart%
1403     \fi%
1404 }%
1405 \newcommand*{\flag@end}{%
1406     \ifledRcol%
1407         \write\linenum@outR{}}%
1408     \else%
1409         \write\linenum@out{}}%
1410     \fi}%
1411 %
1412 %

```

`\ifnumberline` The `\ifnumberline` option can be set to FALSE to disable line numbering.

```

1413 \newif\ifnumberline
1414 \numberlinetrue
1415 %

```

`\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.

```

1416 \newcommand*{\set@line}{%
1417   \ifl@edRcol
1418     \ifx\line@listR\empty
1419       \global\noteschanged@true
1420       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1421     \else
1422       \gl@p\line@listR\to\@tempb
1423       \xdef\l@d@nums{\@tempb|\edfont@info}%
1424       \global\let\@tempb=\undefined
1425     \fi
1426   \else
1427     \ifx\line@list\empty
1428       \global\noteschanged@true
1429       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1430     \else
1431       \gl@p\line@list\to\@tempb
1432       \xdef\l@d@nums{\@tempb|\edfont@info}%
1433       \global\let\@tempb=\undefined
1434     \fi
1435   \fi}
1436
1437 %

```

`\edfont@info` The macro `\edfont@info` returns coded information about the current font.

```

1438 \newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}
1439
1440 %

```

## 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. 112).

```

1441 \newcommand*{\lemma}[1]{%
1442   \global\@lemmacommand@true%
1443   \global\renewcommand{\@tag}{%
1444     \no@expands #1%
1445   }%
1446   \ignorespaces%

```

```
1447 }%
1448 %
```

**\@lemma** The \@lemma is written in the numbered file to set which \edtext has an \lemma as second argument.

```
1449 \newcommand{\@lemma}{%
1450   \booltrue{lemmacommand@the\@edtext@level}%
1451 }%
1452 %
```

**\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.

```
1453 \newif\if@lemmacommand@%
1454 %
```

### 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. 86): 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.

```
1455 \newcommand*{\linenum}[1]{%
1456   \xdef\@tempa{#1|}|}|}|}|}|}\noexpand\\l@d@nums}%
1457   \global\let\l@d@nums=\empty
1458   \expandafter\line@set\@tempa|\\ignorespaces}
1459 %
```

**\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.

```
1460 \def\line@set#1|#2\\#3|#4\\{%
1461   \gdef\@tempb{#1}%
1462   \ifx\@tempb\empty
1463     \l@d@add{#3}%
1464   \else
1465     \l@d@add{#1}%
```

```

1466 \fi
1467 \gdef\@tempb{#4}%
1468 \ifx\@tempb\empty\else
1469   \l@d@add{|\}\line@set#2\#4\}%
1470 \fi}
1471 %

```

`\l@d@add` `\line@set` uses `\l@d@add` to tack numbers or vertical bars onto the right hand end of `\l@d@nums`.

```

1472 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1473
1474 %

```

## 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 `\pstart` and a `\pend`, it does not know yet which are the line numbers.

The general mechanism is the following:

- **At the first run**, each `\sameword` command increments an `etoolbox` counter the name of which contains the argument of the `\sameword` commands.
- Then this counter, associated with the argument of `\sameword` is stored, with the `\@sw` command, in the auxiliary file of the current `eledmac` section (the `.1`, `.2...` file).
- **When this auxiliary file is read at the second run**, different operations are achieved:

1. Get the rank of each `\sameword` in a line (relative rank) from the rank of each `\sameword` in all the numbered section (absolute rank):
  - For each paired `\sameword` argument and absolute line number, a counter is defined. Its value corresponds to the number of times `\sameword{⟨argument⟩}` 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 `\sameword{⟨argument⟩}`.
  - For each `\sameword` having the same argument, we subtract from its absolute rank the number stored for the paired `\sameword` argument and previous absolute line number. Consequently, we obtain the relative rank.
  - See the following example which explain how for same `\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 \sameword by depth of \edtext. That is: create a list for \edtext of level 1, a list for \edtext of level 2, a list for \edtext of level 3 etc. For each \edtext in these list, we store all the relative rank of \saweword 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 \saweword 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>26</sup>

Because there is a bug with `\detokenize` and  $X_{\text{E}}\text{TeX}$  when using non BMP characters<sup>27</sup>, we detokenize only for not  $X_{\text{E}}\text{TeX}$ engines. In any case, in  $X_{\text{E}}\text{TeX}$ , 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.

```

1475 \newcommand{\get@sw@txt}[1]{%
1476   \ifxetex%
1477     \xdef\sw@txt{#1}%
1478   \else%
1479     \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
1480   \fi%
1481 }%
1482 %

```

`\sameword` The high level macro `\sameword`, used by the editor.

```

1483 \newcommand{\sameword}[2][1,usedefault]{%
1484   \leavevmode%
1485   \get@sw@txt{#2}%
1486 %

```

Now, the real code. First, increment the counter corresponding to the argument.

```

1487 \unless\ifledRcol%
1488   \csnumgdef\sw@\sw@txt{\csuse\sw@\sw@txt+1}%
1489 %

```

<sup>26</sup>See <http://tex.stackexchange.com/q/244538/7712>.

<sup>27</sup><http://sourceforge.net/p/xetex/bugs/108/>

Then, write its value to the numbered file.

```
1490 \protected@write\linenum@out{}\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
1491 %
```

Do the same thing if we are in the right columns.

```
1492 \else%
1493 \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1494 \protected@write\linenum@outR{}\string\@sw{\sw@txt}{\csuse{sw@\sw@txt
}}{#1}}%
1495 \fi%
1496 %
```

And print the word.

```
1497 #2%
1498 }%
1499 %
```

A flag set to true if a \@sw relative rank must be added to the list of ranks for a specific \edtext.

```
\if@addsw00 \newif\if@addsw%
1501 %
```

**\@sw** The command printed in the auxiliary files.

```
1502 \newcommand{\@sw}[3]{%
1503 \get@sw@txt{#1}%
1504 \unless\ifledRcol%
1505 %
```

First, define a counter which store the second argument as value for a each paired absolute line number/first argument

```
1506 \csxdef{sw@\sw@txt @\the\absline@num @\the\section@num}{#2}%
1507 %
```

If such argument was not defined for the preceding line, define it.

```
1508 \numdef{\prev@line}{\the\absline@num-1}%
1509 \ifcsundef{sw@\sw@txt @\prev@line @\the\section@num}{%
1510 \csnumgdef{sw@\sw@txt @\prev@line @\the\section@num}{#2-1}%
1511 }{}%
1512 %
```

Then, calculate the position of the word in the line.

```
1513 \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1514 %
```

And do the same thing for the right side.



```

1515 \else%
1516 \csxdef{sw@sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1517 \numdef{\prev@line}{\the\absline@numR-1}%
1518 \ifcsundef{sw@sw@txt @\prev@line @\the\section@numR @R}{%
1519 \csnumgdef{sw@sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1520 }{}%
1521 \numdef{\the@sw}{#2-\csuse{sw@sw@txt @\prev@line @\the\section@numR @R
}}%
1522 \fi%
1523 %

```

And now, add it to the list of \@sw for the current edtext, in all depth.

```

1524 \@tempcnta=\@edtext@level
1525 \@whilenum{\@tempcnta>0}\do{%
1526 \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1527 {%
1528 \@addswfalse%
1529 \notbool{lemmacommand@\the\@tempcnta}%
1530 {\@addswtrue}%
1531 {\IfStrEq{#3}{inlemma}%
1532 {\@addswtrue}%
1533 {%
1534 \def\do##1{%
1535 \ifnumequal{##1}{\the\@tempcnta}%
1536 {\@addswtrue\listbreak}%
1537 }%
1538 }%
1539 \docsvlist{#3}%
1540 }%
1541 }%
1542 \if@addsw%
1543 \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1544 \ifledRcol%
1545 \xright@appenditem{\the@sw}{\the\absline@numR}}\to\@tmp%
1546 \else%
1547 \xright@appenditem{\the@sw}{\the\absline@num}}\to\@tmp%
1548 \fi%
1549 \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1550 \fi%
1551 }%
1552 }%
1553 \advance\@tempcnta by -1%
1554 }%
1555 }%
1556 %

```

`\sameword@inedtext` The command called when `\sameword` is called in a `\edtext`.

```

1557 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1558 \get@sw@txt{#2}%

```

```

1559 \unless\ifledRcol@%
1560 %

```

Just a precaution.

```

1561 \ifx\sw@list@inedtext\empty%
1562 \def\the@sw{999}%
1563 \def\this@absline{-99}%
1564 \else%
1565 %

```

But in many cases, at this step, we should have some content in the list `\sw@list@inedtext`, which contains the reference for `\edtext`.

```

1566 \glp\sw@list@inedtext\to\@tmp%
1567 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1568 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1569 \fi%
1570 %

```

First, calculate the number of occurrences of the word in the current line

```

1571 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
1572 \numdef{\prev@line}{\this@absline-1}%
1573 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@num}-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1574 }%
1575 {\numdef{\sw@atthisline}{0}}%
1576 %

```

Finally, print the rank, but only if there is more than one occurrence of the word in the current line.

```

1577 \ifnumgreater{\sw@atthisline}{1}%
1578 {\showwordrank{#2}{\the@sw}}%
1579 {#2}%
1580 %

```

And the same for right side.

```

1581 \else%
1582 \ifx\sw@list@inedtext\empty%
1583 \def\the@sw{999}%
1584 \def\this@absline{-99}%
1585 \else%
1586 \glp\sw@list@inedtext\to\@tmp%
1587 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1588 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1589 \fi%
1590 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
1591 \numdef{\prev@line}{\this@absline-1}%
1592 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@numR @R}-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1593 }%

```

```

1594         {\numdef{\sw@atthisline}{0}}}%
1595         \ifnumgreater{\sw@atthisline}{1}%
1596             {\showwordrank{#2}{\the@sw}}}%
1597             {#2}%
1598     \fi%
1599 }%
1600 %

```

```

\showwordrank%1 % Finally, the way the rank will be printed.
1602 \newcommand{\showwordrank}[2]{%
1603     #1\textsuperscript{#2}%
1604 }%
1605 %

```

## 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@` When we first form the paragraph, it goes into a box register, `\raw@text`, instead of onto the current vertical list. The `\ifnumberedpar@` flag will be `true` while a paragraph is being processed in that way. `\num@lines` will store the number of lines in the paragraph when it is complete. When we chop it up into lines, each line in turn goes into the `\one@line` register, and `\par@line` will be the number of that line within the paragraph.

```

1606 \newbox\raw@text
1607 \newif\ifnumberedpar@
1608 \newcount\num@lines
1609 \newbox\one@line
1610 \newcount\par@line
1611 %

```

`\pstart` `\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant variables, and then arranges for the subsequent text to go into the `\raw@text` box.

`\AtEveryPstart` `\pstart` needs to appear at the start of every paragraph that is to be numbered; the `\autopar` command below may be used to insert these commands automatically.

`\numberpstarttrue`

`\numberpstartfalse`

`\labelpstarttrue`

`\labelpstartfalse`

`\thepstart`

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

```

1612 \newcommand{\AtEveryPstart}[1]{%
1613   \ifstrempy{#1}%
1614     {\xdef\at@every@pstart{}}%
1615     {\gdef\at@every@pstart{\noindent#1}}%
1616   }%
1617 \xdef\at@every@pstart{}%
1618
1619 \newcounter{pstart}
1620 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
1621 \newif\ifnumberpstart
1622 \numberpstartfalse
1623 \newif\iflabelpstart
1624 \labelpstartfalse
1625 \newcommandx*\pstart[1][1]{%
1626   \normal@pars%
1627   \ifstrempy{#1}{\at@every@pstart}{\noindent#1}%
1628   \ifautopar%
1629     \autopar%
1630   \fi%
1631   \ifluatex%
1632     \edef\l@luatextextdir@L{\the\textdir}%
1633   \fi%
1634   \if@nobreak%
1635     \let\@oldnobreak\@nobreaktrue%
1636   \else%
1637     \let\@oldnobreak\@nobreakfalse%
1638   \fi%
1639   \@nobreaktrue%
1640   \ifnumbering \else%
1641     \led@err@PstartNotNumbered%
1642     \beginnumbering%
1643   \fi%
1644   \ifnumberedpar@%
1645     \led@err@PstartInPstart%
1646   \pend%
1647   \fi%
1648   \list@clear{\inserts@list}%
1649   \global\let\next@insert=\empty%
1650   \begingroup\normal@pars%
1651   \global\advance \l@dnumpstartsL\@ne
1652   \global\setbox\raw@text=\vbox\bgroup%
1653     \ifautopar\else%
1654     \ifnumberpstart%
1655       \ifinstanza\else%
1656       \ifsidepstartnum\else%

```

```

1658     \thepstart%
1659     \fi%
1660     \fi%
1661     \fi%
1662     \fi%
1663     \numberedpar@true%
1664     \iflabelpstart\protected@edef\@currentlabel%
1665         {\p@pstart\thepstart}
1666     \fi%
1667     \l@dzeropenalties%
1668     \ignorespaces%because not automatically ignored if an optional argument
is used (classical TeX behavior for space after commands)
1669 }
1670 %

```

**\pend** \pend must be used to end a numbered paragraph.

```

1671 \newcommand*{\pend}[1][1]{\ifnumbering \else%
1672     \led@err@PendNotNumbered%
1673     \fi%
1674     \global\l@dskipversenumberfalse%
1675     \ifnumberedpar@ \else%
1676         \led@err@PendNoPstart%
1677     \fi%
1678 %

```

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.

```

1679     \l@dzeropenalties%
1680     \endgraf\global\num@lines=\prevgraf\egroup%
1681     \global\par@line=0%
1682 %

```

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.

```

1683     \csnumdef{pstartline}{0}%
1684     \loop\ifvbox\raw@text%
1685         \csnumdef{pstartline}{\pstartline+1}%
1686         \do@line%
1687         \ifbypstart@%
1688             \ifnumequal{pstartline}{1}{%
1689                 \bgroup%
1690                 \let\leavevmode\relax%

```

```

1691     \setline{1}%
1692     \egroup%
1693     \resetprevline@-{}%
1694     \fi%
1695     \repeat%
1696 %

```

Deal with any leftover notes, and then end the group that was begun in the `\pstart`.

```

1697     \flush@notes%
1698     \endgroup%
1699     \ignorespaces%
1700 %

```

Increase `pstart` counter.

```

1701     \ifnumberpstart%
1702         \pstartnumtrue%
1703     \fi%
1704     \addtocounter{pstart}{1}%
1705 %

```

Restore paragraph, nobreak setting and autopar setting.

```

1706     \normal@pars%
1707     \@oldnobreak%
1708     \ifautopar%
1709         \autopar%
1710     \fi%
1711 %

```

Print the optional argument of `\pend` or the content printed after every `\pend`

```

1712     \ifstrempy{#1}{\at@every@pend}{\noindent#1}%
1713 }
1714
1715 %

```

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.

```

\AtEveryPend16
\at@every@pend17 \newcommand{\AtEveryPend}[1]{%
1718     \ifstrempy{#1}%
1719         {\xdef\at@every@pend{}}%
1720         {\gdef\at@every@pend{\noindent#1}}%
1721 }%
1722 \xdef\at@every@pend{}%
1723
1724 %

```

`\l@dzeroopenalties` A macro to zero penalties for `\pend` or `\pstart`.

```

1725 \newcommand*{\l@dzzeropenalties}{%
1726   \brokenpenalty \z@ \clubpenalty \z@
1727   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1728   \postdisplaypenalty \z@ \widowpenalty \z@}
1729
1730 %

```

**\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.

```

1731 \newif\ifautopar
1732 \autoparfalse
1733 \newcommand*{\autopar}{
1734   \ifledRcol
1735     \ifnumberingR \else
1736       \led@err@AutoparNotNumbered
1737       \beginnumberingR
1738       \fi
1739     \else
1740       \ifnumbering \else
1741         \led@err@AutoparNotNumbered
1742         \beginnumbering
1743         \fi
1744       \fi
1745     \autopartrue
1746     \everypar{\setbox0=\lastbox
1747       \endgraf \vskip-\parskip

```

```

1748 \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\
fi\fi
1749 \let\par=\pend}%
1750 \ignorespaces}
1751 %

```

**\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.

```

1752 \newcommand*{\normal@pars}{\everypar{}\let\par\endgraf}
1753
1754 %

```

**\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.

```

1755 \newif\ifautopar@pause
1756 %

```

## 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**

```

1757 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1758 \newcommand*{\do@line}{%
1759   {\vbadness=10000
1760     \splittopskip=\z@
1761     \do@linehook
1762   \l@demptyd@ta
1763     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1764   \unvbox\one@line \global\setbox\one@line=\lastbox
1765   \getline@num
1766   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{\}
1767   \ifnum\@lock>\@one
1768     \inserthangingsymboltrue
1769   \else
1770     \inserthangingsymbolfalse
1771   \fi
1772   \check@pb@in@verse
1773   \ifl@dhidenumber%
1774     \global\l@dhidenumberfalse%
1775     \f@x@l@cks%
1776   \else%
1777     \affixline@num%
1778   \fi%
1779 %

```



Depending whether a sectioning command is called at this pstart or not we print sectioning command or normal line,

```

1780 \xifinlist{\the\l@dumpstartsL}{\eled@sections@@}%
1781     {\print@eledsection}%
1782     {\print@line}%
1783 \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{\}
1784 }%
1785 %

```

### VII.2.2 Process for “normal” line

`\print@line` `\print@line` is for normal line, i. e. line without sectioning command.

```

1786 \def\print@line{
1787 %

```

Insert the pstart number in side, if we are in the first line of a pstart.

```

1788     \affixpstart@num%
1789 %

```

The line will be boxed, to have the good width.

```

1790     \hb@xt@ \linewidth{\%
1791 %

```

User hook.

```

1792     \do@insidelinehook%
1793 %

```

Left line number

```

1794     \l@dld@ta%
1795 %

```

Restore marginal and footnotes.

```

1796     \add@inserts\affixside@note%
1797 %

```

Print left notes.

```

1798     \l@dlsn@te
1799 %

```

Boxes the line, writes information about new line in the numbered file.

```

1800     {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1801 %

```

If we use Lua<sub>TEX</sub> then restore the direction.

```

1802     \ifluatex%
1803     \texdir\l@luatextextdir@L%
1804     \fi%
1805 %

```

Insert, if needed, the hanging symbol.

```
1806 \inserthangingsymbol %Space keep for backward compatibility
1807 %
```

And so, print the line.

```
1808 \l@dunhbox@line{\one@line}}%
1809 %
```

Right line number

```
1810 \ledrlfill\l@drd@ta%
1811 %
```

Print right notes.

```
1812 \l@drsn@te
1813 }}%
1814 %
```

And reinsert penalties (for page breaking)...

```
1815 \add@penalties%
1816 }
1817 %
```

### 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.

```
1818 \def\print@eledsection{%
1819 \add@inserts\affixside@note%
1820 \numdef{\temp@}{\l@dnumpstartsL-1}%
1821 \xifinlist{\temp@}{\eled@sections@@}{\@nobreaktrue}{\@nobreakfalse}%
1822 \@eled@sectioningtrue%
1823 \csuse{eled@sectioning@the\l@dnumpstartsL}%
1824 \@eled@sectioningfalse%
1825 \global\csundef{eled@sectioning@the\l@dnumpstartsL}%
1826 \if@RTL%
1827 \hspace{-3\paperwidth}%
1828 {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1829 \else%
1830 \hspace{3\paperwidth}%
1831 {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1832 \fi%
1833 \vskip-\baselineskip%
1834 }
1835 %
```

### VII.2.4 Hooks

`\do@linehook` 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.

```
1836 \newcommand*{\do@linehook}{}
1837 \newcommand*{\do@insidelinehook}{}
1838 %
```

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

```
1839 \newcommand*{\dolinehook}[1]{\gdef\do@linehook{#1}}%
1840 \newcommand*{\doinsidelinehook}[1]{\gdef\do@insidelinehook{#1}}%
1841
1842 %
```

### VII.2.5 Sidenotes and marginal line number initialization

`\l@emptyd@ta` Nulls the `\. . .d@ta`, which may later hold line numbers. Similarly for `\l@dcsnotetext`, `\l@dld@ta` `\l@dcsnotetext@l`, `\l@dcsnotetext@r` for the texts of the sidenotes, left and right notes.

```
1843 \l@dcsnotetext
1844 \l@dcsnotetext@l
1845 \l@dcsnotetext@r
1846 \newcommand*{\l@emptyd@ta}{%
1847 \gdef\l@dld@ta{}%
1848 \gdef\l@drd@ta{}%
1849 \gdef\l@dcsnotetext@l{}%
1850 \gdef\l@dcsnotetext@r{}%
1851 \gdef\l@dcsnotetext{}%
1852 %
```

`\l@dlsn@te` Zero width boxes of the left and right side notes, together with their kerns.

```
1851 \l@dlsn@te
1852 \l@dlsn@te
1853 \newcommand*{\l@dlsn@te}{%
1854 \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1855 \newcommand*{\l@dlsn@te}{%
1856 \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1857 %
```

`\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`.

```
1857 \newcommand*{\ledllfill}{\hfil}
1858 \newcommand*{\ledrlfill}{}
1859
1860 %
```

## 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.

```

1861 \newcommand*{\getline@num}{%
1862   \global\advance\absline@num \@ne%
1863   \do@actions
1864   \do@ballast
1865   \ifnumberline
1866     \ifsublines@
1867       \ifnum\sub@lock<\tw@
1868         \global\advance\subline@num \@ne
1869       \fi
1870     \else
1871       \ifnum\@lock<\tw@
1872         \global\advance\line@num \@ne
1873         \global\subline@num \z@
1874       \fi
1875     \fi
1876   \fi
1877 }
1878 %

```

`\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. 141).

`\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.

```

1879 \newcount\ballast@count
1880 \newcounter{ballast}
1881 \setcounter{ballast}{0}
1882 %

```

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.

```

1883 \newcommand*{\do@ballast}{\global\ballast@count \z@
1884   \begingroup
1885     \advance\absline@num \@ne
1886     \ifnum\next@actionline=\absline@num
1887       \ifnum\next@action>-1001\relax
1888         \global\advance\ballast@count by -\c@ballast
1889       \fi
1890     \fi

```

```

1891 \endgroup}
1892 %

```

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

**\do@actions@next**

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`.

```

1893 \newcommand*{\do@actions}{%
1894   \global\let\do@actions@next=\relax
1895   \ifnum\absline@num<\next@actionline\else
1896   %

```

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.

```

1897   \ifnum\next@action>-1001
1898     \global\page@num=\next@action
1899     \ifbypage@
1900       \global\line@num=\z@ \global\subline@num=\z@
1901       \resetprevline@
1902     \fi
1903   %

```

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`.)

```

1904   \else
1905     \ifnum\next@action<-4999
1906       \@l@tempcnta=-\next@action
1907       \advance\@l@tempcnta by -5001
1908       \ifsublines@
1909         \global\subline@num=\@l@tempcnta
1910       \else
1911         \global\line@num=\@l@tempcnta
1912       \fi
1913   %

```

We rescale the value in `\@l@tempcnta` so that we can use a case statement.

```

1914   \else
1915     \@l@tempcnta=-\next@action
1916     \advance\@l@tempcnta by -1000
1917     \do@actions@fixedcode
1918   \fi
1919 \fi
1920 %

```

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.

```

1921 \ifx\actionlines@list\empty
1922   \gdef\next@actionline{1000000}%
1923 \else
1924   \gl@p\actionlines@list\to\next@actionline
1925   \gl@p\actions@list\to\next@action
1926   \global\let\do@actions@next=\do@actions
1927 \fi
1928 \fi
1929 %

```

Make the recursive call, if necessary.

```

1930 \do@actions@next}
1931
1932 %

```

`\do@actions@fixedcode` This macro handles the fixed codes for `\do@actions`. It is one big case statement.

```

1933 \newcommand*{\do@actions@fixedcode}{%
1934 \ifcase\@l@dttempcnta
1935 \or% % 1001
1936   \global\sublines@true
1937 \or% % 1002
1938   \global\sublines@false
1939 \or% % 1003
1940   \global\@lock=\@ne
1941 \or% % 1004
1942   \ifnum\@lock=\tw@
1943     \global\@lock=\thr@@
1944   \else
1945     \global\@lock=\z@
1946   \fi
1947 \or% % 1005
1948   \global\sub@lock=\@ne
1949 \or% % 1006
1950   \ifnum\sub@lock=\tw@
1951     \global\sub@lock=\thr@@
1952   \else
1953     \global\sub@lock=\z@
1954   \fi
1955 \or% % 1007
1956   \l@dskipnumbertrue
1957 \or% % 1008
1958   \l@dskipversenumbertrue%
1959 \or% % 1009
1960   \l@dhidnumbertrue

```

```

1961 \else
1962   \led@warn@BadAction
1963 \fi}
1964
1965
1966 %

```

## 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:

$$n = \text{int}((\text{linenum} - \text{firstlinenum}) / \text{linenumincrement})$$

$$m = \text{firstlinenum} + (n \times \text{linenumincrement})$$

(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 ≤ \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.

```

1967 \newcommand*{\affixline@num}{%
1968 %

```

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).

```

1969 \ifledgroupnotesL@else
1970   \ifnumberline
1971     \ifl@dskipnumber
1972       \global\l@dskipnumberfalse
1973     \else
1974       \ifsublines@
1975         \@l@tempcntb=\subline@num
1976         \ifnum\subline@num>\c@firstsublinenum
1977           \@l@tempcnta=\subline@num
1978           \advance\@l@tempcnta by-\c@firstsublinenum
1979           \divide\@l@tempcnta by\c@sublinenumincrement
1980           \multiply\@l@tempcnta by\c@sublinenumincrement
1981           \advance\@l@tempcnta by\c@firstsublinenum
1982         \else
1983           \@l@tempcnta=\c@firstsublinenum
1984         \fi
1985       %

```

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.

```
1986 \ch@cksub@l@ck
1987 %
```

Now the line number case, which works the same way.

```
1988 \else
1989 \@l@dttempcntb=\line@num
1990 %
```

Check on the `\linenumberlist` If it is `\empty` use the standard algorithm.

```
1991 \ifx\linenumberlist\empty
1992 \ifnum\line@num>\c@firstlinenum
1993 \@l@dttempcnta=\line@num
1994 \advance\@l@dttempcnta by-\c@firstlinenum
1995 \divide\@l@dttempcnta by\c@linenumincrement
1996 \multiply\@l@dttempcnta by\c@linenumincrement
1997 \advance\@l@dttempcnta by\c@firstlinenum
1998 \else
1999 \@l@dttempcnta=\c@firstlinenum
2000 \fi
2001 \else
2002 %
```

The `\linenumberlist` was not `\empty`, so here is Wayne's numbering mechanism. This takes place in  $\TeX$ 's mouth.

```
2003 \@l@dttempcnta=\line@num
2004 \edef\rem@inder{\,\linenumberlist,\number\line@num,}%
2005 \edef\sc@n@list{\def\noexpand\sc@n@list
2006 ###1,\number\@l@dttempcnta,###2|\def\noexpand\rem@inder
2007 {####2}}}%
2008 \sc@n@list\expandafter\sc@n@list\rem@inder|}%
2009 \ifx\rem@inder\empty%
2010 \advance\@l@dttempcnta\@ne
2011 \fi
2012 %
```

A locking check for lines, just like the version for sub-line numbers above.

```
2013 \ch@ck@l@ck
2014 \fi
2015 %
```

The following tests are true if we need to print a line number.

```
2016 \ifnum\@l@dttempcnta=\@l@dttempcntb
2017 \ifl@dskipversenumber\else
2018 %
```



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  $\text{\LaTeX}$  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`.  
`\l@drd@ta`

```

2019         \if@twocolumn
2020             \if@firstcolumn
2021                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2022             \else
2023                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2024             \fi
2025         \else
2026             \l@dttempcntb=\line@margin
2027             \ifnum\l@dttempcntb>\@ne
2028                 \advance\l@dttempcntb \page@num
2029             \fi
2030             \ifodd\l@dttempcntb
2031                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2032             \else
2033                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2034             \fi
2035         \fi
2036     \fi
2037 \fi
2038 %

```

Now fix the lock counters, if necessary. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

2039         \f@x@l@cks
2040     \fi
2041 \fi
2042 \fi
2043 }
2044 %
2045

```

`\ch@cksub@l@ck` These macros handle line number locking for `\affixline@num`. `\ch@cksub@l@ck`  
`\ch@ck@l@ck` checks subline locking. If it fails, then we disable the line-number display by setting the  
`\f@x@l@cks` counters to arbitrary but unequal values.

```

2046 \newcommand*{\ch@cksub@l@ck}{%
2047   \ifcase\sub@lock
2048     \or
2049       \ifnum\sublock@disp=\@ne
2050         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2051       \fi
2052     \or
2053       \ifnum\sublock@disp=\tw@ \else
2054         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2055       \fi
2056     \or
2057       \ifnum\sublock@disp=\z@
2058         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2059       \fi
2060   \fi}
2061 %

```

Similarly for line numbers.

```

2062 \newcommand*{\ch@ckl@ck}{%
2063   \ifcase\@lock
2064     \or
2065       \ifnum\lock@disp=\@ne
2066         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2067       \fi
2068     \or
2069       \ifnum\lock@disp=\tw@ \else
2070         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2071       \fi
2072     \or
2073       \ifnum\lock@disp=\z@
2074         \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2075       \fi
2076   \fi}
2077 %

```

Fix the lock counters. A value of 1 is advanced to 2; 3 advances to 0; other values are unchanged.

```

2078 \newcommand*{\f@x@l@cks}{%
2079   \ifcase\@lock
2080     \or
2081       \global\@lock=\tw@
2082     \or \or
2083       \global\@lock=\z@
2084     \fi
2085   \ifcase\sub@lock
2086     \or
2087       \global\sub@lock=\tw@
2088     \or \or
2089       \global\sub@lock=\z@
2090   \fi}

```

```
2091
2092 %
```

## 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.

```
\leftpstartnum 93
\rightpstartnum 94 \newif\ifsidepstartnum
\ifsidepstartnum 95 \newcommand*{\affixpstart@num}{%
2096   \ifsidepstartnum
2097     \if@twocolumn
2098       \if@firstcolumn
2099         \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
2100       \else
2101         \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
2102       \fi
2103     \else
2104       \@l@tempcntb=\line@margin
2105       \ifnum\@l@tempcntb>\@ne
2106         \advance\@l@tempcntb \page@num
2107       \fi
2108       \ifodd\@l@tempcntb
2109         \gdef\l@drd@ta{\rlap{{\rightpstartnum}}}%
2110       \else
2111         \gdef\l@dld@ta{\llap{{\leftpstartnum}}}%
2112       \fi
2113     \fi
2114   \fi
2115 }
2116 %
2117
2118
2119 \newif\ifpstartnum
2120 \pstartnumtrue
2121 \newcommand*{\leftpstartnum}{
2122   \ifpstartnum\thepstart
2123   \kern\linenumsep\fi
```

```

2124 \global\pstartnumfalse
2125 }
2126 \newcommand*{\rightpstartnum}{
2127   \ifpstartnum
2128     \kern\linenumsep
2129     \thepstart
2130     \fi
2131   \global\pstartnumfalse
2132 }
2133 %

```

## 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.

```

2134 \list@create{\inserts@list}
2135 %

```

`\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`.

```

2136 \newcommand*{\add@inserts}{%
2137   \global\let\add@inserts@next=\relax
2138   %

```

If `\inserts@list` is empty, there are not any more notes or insertions for this paragraph, and we need not waste our time.

```

2139   \ifx\inserts@list\empty \else
2140   %

```

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.

```

2141 \ifx\next@insert\empty
2142   \ifx\insertlines@list\empty
2143     \global\noteschanged@true
2144     \gdef\next@insert{100000}%
2145   \else
2146     \gl@p\insertlines@list\to\next@insert
2147   \fi
2148 \fi
2149 %

```

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 ourself recursively: there might be another insert for this same line.

```

2150 \ifnum\next@insert=\absline@num
2151   \gl@p\inserts@list\to\@insert
2152   \@insert
2153   \global\let\@insert=\undefined
2154   \global\let\next@insert=\empty
2155   \global\let\add@inserts@next=\add@inserts
2156 \fi
2157 \fi
2158 %

```

Make the recursive call, if necessary.

```

2159 \add@inserts@next}
2160
2161 %

```

## 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@dttempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast` above (VIII p. 132). Finally, the penalty is checked to see that it does not go below  $-10000$ .

```

2162 \newcommand*{\add@penalties}{\@l@dttempcnta=\ballast@count
2163   \ifnum\num@lines>\@ne
2164     \global\advance\par@line \@ne
2165     \ifnum\par@line=\@ne

```

```

2166     \advance\@l@dttempcnta \clubpenalty
2167     \fi
2168     \@l@dttempcntb=\par@line \advance\@l@dttempcntb \@ne
2169     \ifnum\@l@dttempcntb=\num@lines
2170         \advance\@l@dttempcnta \widowpenalty
2171     \fi
2172     \ifnum\par@line<\num@lines
2173         \advance\@l@dttempcnta \interlinepenalty
2174     \fi
2175 \fi
2176 \ifnum\@l@dttempcnta=\z@
2177     \relax
2178 \else
2179     \ifnum\@l@dttempcnta>-10000
2180         \penalty\@l@dttempcnta
2181     \else
2182         \penalty -10000
2183     \fi
2184 \fi}
2185
2186 %

```

### 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.

```

2187 \newcommand*{\flush@notes}{%
2188     \@xloop
2189     \ifx\inserts@list\empty \else
2190         \gl@p\inserts@list\to\@insert
2191         \@insert
2192         \global\let\@insert=\undefined
2193     \repeat}
2194
2195 %

```

**\@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 Kabsch in *TUGboat* 8 (1987), pp. 184–5.

```
2196 \def\@xloop#1\repeat{%
2197   \def\body{#1\expandafter\body\fi}%
2198   \body}
2199
2200 %
```

## 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` `\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.

```
2201 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@@lemmafont#7|}
2202 \def\select@@lemmafont#1/#2/#3/#4|%
2203   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2204   \selectfont}
2205
2206 %
```

### 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.

```
2207 \newcommand*\footnoteoptions@[3]{%
2208   \def\do##1{%
2209     \ifstrequal{#1}{L}{% In Leftside
2210       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
unexpanded{##1}}}\to\inserts@list%
2211       \global\advance\insert@count \@one% Increment the left insert
counter.
2212     }%
}
```

```

2213     {%
2214         \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
unexpanded{##1}}}\to\inserts@listR%
2215         \global\advance\insert@countR \@ne% Increment the right insert
counter insert.
2216     }%
2217 }%
2218 \notblank{#2}{\docsvlist{#2}}}% Parsing all options
2219 }
2220 %

```

### 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.

```

2221 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2222     \ifstrequal{#1}{L}{%
2223         \xright@appenditem{{\csxdef{footnote@luatextextdir}{\the\textdir}}}\to\
inserts@list%Know the dir of lemma
2224         \global\advance\insert@count \@ne%
2225         \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\pardir}}}\to\
inserts@list%Know the dir of lemma
2226         \global\advance\insert@count \@ne%
2227     }%
2228     {%
2229         \xright@appenditem{{\csxdef{footnote@luatextextdir}{\the\textdir}}}\to\
inserts@listR%Know the dir of lemma
2230         \global\advance\insert@countR \@ne%
2231         \xright@appenditem{{\csxdef{footnote@luatexpardir}{\the\pardir}}}\to\
inserts@listR%Know the dir of lemma
2232         \global\advance\insert@countR \@ne%
2233     }%
2234 }
2235 %

```

`\footnotelang@poly` `\footnotelang@poly` is called to remember the information about the language of a lemma when polyglossia is used.

```

2236 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2237     \ifstrequal{#1}{L}{%
2238         \if@RTL%
2239             \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@list%Know the language used in the lemma
2240             \global\advance\insert@count \@ne%
2241         \else
2242             \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@list%Know the language of lemma
2243             \global\advance\insert@count \@ne%

```



```

2244 \fi%
2245 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@list%Know the language of lemma
2246 \global\advance\insert@count \@ne%
2247 }%
2248 {%
2249 \if@RTL
2250 \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
2251 \global\advance\insert@countR \@ne%
2252 \else
2253 \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2254 \global\advance\insert@countR \@ne%
2255 \fi
2256 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@listR%Know the language of lemma
2257 \global\advance\insert@countR \@ne%
2258 }%
2259 }
2260 %

```

## 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 `\vAfootnote`, `\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  $\text{\LaTeX}$  `\floatingpenalty`.

```

2261 \newcommand*{\footplitskips}{%
2262   \interlinepenalty=\interfootnotelinepenalty
2263   \unless\ifl@dprintingpages%
2264     \floatingpenalty=\@MM%
2265   \fi%
2266   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2267   \leftskip=\z@skip \rightskip=\z@skip}
2268
2269 %

```

`\normalfootnoterule` `\normalfootnoterule` is a standard footnote-rule macro, for use by a `footstart` macro: just the same as the PLAIN  $\text{\TeX}$  footnote rule.

```

2270 \let\normalfootnoterule=\footnoterule
2271 %

```

## XII.6 Footnotes arrangement

### XII.6.1 User level macro

`\Xarrangement` `\Xarrangement[ $\langle s \rangle$ ]{ $\langle arrangement \rangle$ }` The command calls, for each series, a specific command which set many counters and commands in order to define specific arrangement.

```

2272 \newcommandx{\Xarrangement}[2][1,usedefault]{%
2273   \def\do##1{%
2274     \csname Xarrangement@#2\endcsname{##1}%
2275   }%
2276   \ifstrempy{#1}%
2277     {%
2278       \dolistloop{\@series}%
2279     }%
2280     {
2281       \docsvlist{#1}%
2282     }%
2283   }%
2284 %

```

### 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\vsiz
\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.

```
2285 \newcommand*{\Xarrangement@normal}[1]{%
2286   \csdef{series@display#1}{normal}
2287   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2288   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2289   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
2290   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2291   \expandafter\let\csname #1footnoterule\endcsname=%
2292                                     \normalfootnoterule
2293   \count\csname #1footins\endcsname=1000
2294   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2295   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2296   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2297   %
```

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.

```
2298   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
2299   side only
2300   %
```

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).

```
2300 \ifnoledgroup@else%
2301   \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2302   \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2303   \count\csname mp#1footins\endcsname=1000
```

```

2304 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2305 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2306 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2307 \fi
2308 }
2309
2310 %

```

**\normalvfootnote** We now begin a series of commands that do ‘normal’ footnote formatting: a format much like that implemented in PLAIN T<sub>E</sub>X, 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.

```

2311 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
2312 \insert\csname #1footins\endcsname\bgroup
2313 \hsize=\expandafter\dimexpr\csuse{Xhsize@#1}\relax%
2314 \noindent\csuse{Xhooknote@#1}%
2315 \csuse{Xnotefontsize@#1}%
2316 \footsplitskips
2317 \ifl@dpairing\ifl@dpadding\else%
2318 \setXnoteswidthliketwocolumns@{#1}%
2319 \fi\fi%
2320 \setXnotespositionliketwocolumns@{#1}%
2321 \spaceskip=\z@skip \xspaceskip=\z@skip
2322 \csname #1footfmt\endcsname #2{#1}\egroup}
2323 %

```

**\mpnormalvfootnote** And a somewhat different version for minipages.

```

2324 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2325 \global\setbox\@nameuse{mp#1footins}\vbox{%
2326 \unvbox\@nameuse{mp#1footins}
2327 \noindent\csuse{Xhooknote@#1}%
2328 \csuse{Xnotefontsize@#1}%
2329 \hsize\columnwidth
2330 \@parboxrestore
2331 \color@begingroup
2332 \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2333
2334 %

```

**\normalfootfmt** **\normalfootfmt** is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 86), 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.

```

2335
2336
2337 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalfootfmt}[4]{%
2338   \Xledsetnormalparstuff{#4}%
2339   \hangindent=\csuse{Xhangindent@#4}
2340   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2341   \strut{\printlinefootnote{#1}{#4}}%
2342   {\nottoggle{Xlemmadisablefontselection@#4}%
2343     {\select@lemmafont#1|{\csuse{Xlemmafont@#4}#2}}%
2344     {\csuse{Xlemmafont@#4}#2}}%
2345   }%
2346   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2347     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2348     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2349   }}%
2350   #3\strut\par}
2351 %

```

**\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 \skip\Xfootins value for the associated series of notes. T<sub>E</sub>X 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.

```

2352 \newcommand*{\normalfootstart}[1]{%
2353 %

```

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.

```

2354 \ifdimequal{0pt}{\preXnotes@}{%
2355   {%
2356   \iftoggle{preXnotes@}{%
2357     \togglefalse{preXnotes@}%

```

```

2358         \skip\csname #1footins\endcsname=%
2359         \dimexpr\csuse{preXnotes@}+\csuse{Xafterterrule@#1}\relax%
2360         }%
2361     {}%
2362 }%
2363 \vskip\skip\csname #1footins\endcsname%
2364 %

```

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.

```

2365 \leftskipOpt \rightskipOpt
2366 \ifl@dpairing\else%
2367     \hsize=\old@hsize%
2368 \fi%
2369 \setXnoteswidthliketwocolumns@{#1}%
2370 \setXnotespositionliketwocolumns@{#1}%
2371 %

```

And now, print the footnote's rule to finish the footnote's introduction.

```

2372 \print@Xfootnoterule{#1}%
2373 \noindent\leavevmode}
2374 %

```

`\normalfootgroup` `\normalfootgroup` is a standard footnote-grouping macro: it sends the contents of the footnote-insert box to the output page without alteration.

```

2375 \newcommand*{\normalfootgroup}[1]{%
2376     {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2377     \csuse{Xbhookgroup@#1}%
2378     \unvbox\csname #1footins\endcsname%
2379     \hsize=\old@hsize%
2380 }%
2381
2382 %

```

`\mpnormalfootgroup` A somewhat different version for minipages. Note that, in this case, we do not make distinctions between the `\Xfootgroup` and `\Xfootstarts` macros.

```

2383 \unless\ifnoledgroup@
2384 \newcommand*{\mpnormalfootgroup}[1]{%
2385     \vskip\skip\@nameuse{mp#1footins}
2386     \ifl@dpairing\ifparledgroup%
2387         \leavevmode\marks\parledgroup@{begin}%
2388         \marks\parledgroup@series{#1}%
2389         \marks\parledgroup@type{Xfootnote}%
2390     \fi\fi\normalcolor%
2391     \ifparledgroup%
2392         \ifl@dpairing%

```

```

2393 \else%
2394 \setXnoteswidthliketwocolumns@{#1}%
2395 \setXnotespositionliketwocolumns@{#1}%
2396 \print@Xfootnoterule{#1}%%
2397 \fi%
2398 \else%
2399 \setXnoteswidthliketwocolumns@{#1}%
2400 \setXnotespositionliketwocolumns@{#1}%
2401 \print@Xfootnoterule{#1}%%
2402 \fi%
2403 \setlength{\parindent}{0pt}
2404 {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2405 \csuse{Xbhookgroup@#1}%
2406 \unvbox\csname mp#1footins\endcsname}}
2407 \fi
2408 %

```

### XII.6.3 Paragraphed footnotes

The paragraphed-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 paragraphed; 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.

```

2409 \newcommand*{\Xarrangement@paragraph}[1]{%
2410 \csgdef{series@display#1}{paragraph}
2411 \expandafter\newcount\csname #1prevpage@num\endcsname
2412 \expandafter\let\csname #1footstart\endcsname=\parafootstart
2413 \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2414 \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2415 \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2416 \count\csname #1footins\endcsname=1000
2417 \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2418 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2419 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2420 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2421 \para@footsetup{#1}
2422 %

```

And the extra setup for minipages.

```

2423 \ifnoledgroup@else
2424   \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2425   \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2426   \count\csname mp#1footins\endcsname=1000
2427   \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2428   \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2429   \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2430 \fi
2431 }
2432 %

```

**\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.

```

2433 \providecommand{\footfudgefiddle}{64}
2434 %

```

**\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`.

```

2435 \newcommand*{\para@footsetup}[1]{\csuse{Xbhookgroup@#1}\csuse{
Xnotefontsize@#1}
2436   \setXnoteswidthliketwocolumns@{#1}%
2437   \ifcempty{Xhsize@#1}%
2438     {}%
2439     {\columnwidth=\expandafter\dimexpr\csuse{Xhsize@#1}\relax}%
2440   \dimen0=\baselineskip
2441   \multiply\dimen0 by 1024
2442   \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2443   \csxdef{#1footfudgefactor}{%
2444     \expandafter\strip@pt\dimen0 }}
2445
2446 %

```

`\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.



```

2447 \newcommand*{\parafootstart}[1]{%
2448   \rightskip=0pt \leftskip=0pt%
2449   \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
2450   \ifdimequal{0pt}{\preXnotes@}{}%
2451   {%
2452     \iftoggle{preXnotes@}{%
2453       \togglefalse{preXnotes@}%
2454       \skip\csname #1footins\endcsname=%
2455       \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2456     }%
2457   }%
2458 }%
2459 \vskip\skip\csname #1footins\endcsname%
2460 \setXnoteswidthliketwocolumns@{#1}%
2461 \setXnotespositionliketwocolumns@{#1}%
2462 \print@Xfootnoterule{#1}%
2463 \let\bidirTL@everypar\@empty%
2464 \noindent\leavevmode}
2465 %

```

**\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>28</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>29</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

<sup>28</sup>Michael Downes, ‘Line Breaking in `\unhboxed` Text’, *TUGboat* **11** (1990), pp. 605–612.

<sup>29</sup>See *The TeXbook*, p. 455 (editions after January 1990).

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>30</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>31</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. 149 above). We need to do this, since `\footfudgefactor` is calculated on the assumption that the notes are `\hsize` 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.

```

2466 \newcommand*{\paravfootnote}[2]{%
2467   \insert\csname #1footins\endcsname
2468   \bgroup
2469     \csuse{Xnotefontsize@#1}
2470     \footplitskips
2471     \setbox0=\vbox{\hsize=\maxdimen%
2472       \let\bidir@RTL@everypar@empty%
2473       \noindent\csuse{Xhooknote@#1}%
2474       \csname #1footfmt\endcsname #2{#1}}%
2475     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2476     \dp0=0pt
2477     \ht0=\csname #1footfudgefactor\endcsname\wd0
2478   %

```

Here we produce the contents of the footnote from box 0, and add a penalty of 0 between boxes in this insert.

```

2479   \if@RTL\noindent \leavevmode\fi\box0%
2480   \penalty0
2481 \egroup}
2482 %
2483 %

```

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

<sup>30</sup>Wayne supplied his own macros to do this, but since they were almost identical to Michael’s, Peter Wilson have used the latter’s `\Xunvxh` macro since it is publicly documented.

<sup>31</sup>‘Line Breaking’, p. 610.

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.

```

2484 \newcommand*{\mpparavfootnote}[2]{%
2485   \global\setbox\@nameuse{mp#1footins}\vbox{%
2486     \unvbox\@nameuse{mp#1footins}%
2487     \csuse{Xnotefontsize@#1}
2488     \footssplitsskip
2489     \setbox0=\vbox{\hsize=\maxdimen%
2490       \let\bidir@RTL@everypar\@empty%
2491       \noindent\color@begingroup%
2492       \csuse{Xbhooknote@#1}%
2493       \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2494     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2495     \dp0=\z@
2496     \ht0=\csname #1footfudgefactor\endcsname\wd0
2497     \box0
2498     \penalty0
2499   }}
2500
2501 %

```

`\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  $\TeX$  automatically attaches to the end of paragraphs. When  $\TeX$  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`.

```

2502 \newcommand*{\Xunvxh}[2]{%
2503   \setbox0=\vbox{\unvbox#1%
2504     \global\setbox1=\lastbox}%
2505   \unhbox1
2506   \unskip           % remove \rightskip,
2507   \unskip           % remove \parfillskip,
2508   \unpenalty        % remove \penalty of 10000,
2509   \hskip\csuse{Xafternote@#2} % but add the glue to go between the notes
2510
2511 %

```

`\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).

```

2512 \newcommand*{\parafootfmt}[4]{%
2513   \Xinsertparafootsep{#4}%
2514   \ledsetnormalparstuff@common%
2515   \printlinefootnote{#1}{#4}%
2516   {\nottoggle{Xlemmadisablefontselection@#4}%
2517     {\select@lemmafont#1|{\csuse{Xlemmafont@#4}#2}}}%
2518     {\csuse{Xlemmafont@#4}#2}}}%
2519   }%
2520   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2521     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2522     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2523   }%
2524   #3\penalty-10 }
2525 %

```

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.

```

2526 \newcommand*{\parafootgroup}[1]{%
2527   \hsize=\expandafter\dimexpr\csuse{Xhsize@#1}\relax%
2528   \unvbox\csname #1footins\endcsname
2529   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2530   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2531   \makehboxofhboxes
2532   \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2533   \csuse{Xhookgroup@#1}%
2534   \csuse{Xnotefontsize@#1}%
2535   \unhbox0\par%
2536   \global\hsize=\old@hsize%
2537   }%
2538
2539 %

```

**`\mpparafootgroup`** The minipage version.

```

2540 \newcommand*{\mpparafootgroup}[1]{%
2541   \setXnoteswidthliketwocolumns@{#1}%
2542   \vskip\skip\@nameuse{mp#1footins}
2543   \ifl@dpairing\ifparledgroup%
2544     \leavevmode\marks\parledgroup@{begin}%
2545     \marks\parledgroup@series{#1}%
2546     \marks\parledgroup@type{Xfootnote}%
2547   \fi\fi\normalcolor
2548   \ifparledgroup%
2549     \ifl@dpairing%
2550     \else%
2551       \setXnoteswidthliketwocolumns@{#1}%
2552       \setXnotespositionliketwocolumns@{#1}%
2553       \print@Xfootnoterule{#1}%%
2554     \fi%
2555   \else%
2556     \setXnoteswidthliketwocolumns@{#1}%
2557     \setXnotespositionliketwocolumns@{#1}%
2558     \print@Xfootnoterule{#1}%
2559   \fi%
2560   \unvbox\csname mp#1footins\endcsname
2561   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2562   \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2563   \makehboxofhboxes
2564   \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2565   \csuse{Xhookgroup@#1}%
2566   \csuse{Xnotefontsize@#1}%
2567   \unhbox0\par}}
2568
2569 %

```

And finally, the two macros which are required to transform the long horizontal box stored in the insert' box to a printable text.

```

\makehboxofhboxes 70 \newcommand*{\makehboxofhboxes}{\setbox0=\hbox{}}%
\removehboxes 71   \loop
2572     \unpenalty
2573     \setbox2=\lastbox
2574     \ifhbox2
2575       \setbox0=\hbox{\box2\unhbox0}%
2576     \repeat}
2577
2578 \newcommand*{\removehboxes}{\setbox0=\lastbox
2579   \ifhbox0{\removehboxes}\unhbox0 \fi}
2580
2581 %

```

**Insertion of the footnotes separator** The command `\Xinsertparafootsep{<series>}` must be called at the beginning of `\parafootftm`.

```

\prevpage@num\newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep
    \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
    {\ifcndef{prevline#1}% Be sur \prevline#1 exists.
    {\ifnumequal{\csuse{prevline#1}}{\line@num}%
    {\ifcempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
    {\csuse{Xparafootsep@#1}}%
    }%
    {\csuse{Xparafootsep@#1}}%
    }%
    {}%
    \global\csname #1prevpage@num\endcsname=\page@num%
    }
    %

```

## XII.6.4 Columnar footnotes

### Common tools

`\rigidbalance` 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 `\Xrigidbalance` 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  $\text{\LaTeX}$  `\line` macro has no relationship to the TeX `\line`. The  $\text{\LaTeX}$  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`.

```

2595 \newcount\@k \newdimen\@h
2596 \newcommand*\Xrigidbalance}[3]{%
2597     \hsize=\expandafter\dimexpr\csuse{Xhsize@\@currentseries}\relax%
2598     \rigidbalance{#1}{#2}{#3}%
2599 }%
2600
2601 \newcommand*\rigidbalanceX}[3]{%
2602     \hsize=\expandafter\dimexpr\csuse{hsizeX@\@currentseries}\relax%
2603     \rigidbalance{#1}{#2}{#3}%
2604 }%
2605
2606 \newcommand*\rigidbalance}[3]{%
2607     \setbox0=\box#1 \@k=#2 \@h=#3%
2608     \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
2609     \valign{##\vfil\cr\dosplits}}
2610

```

```

2611 \newcommand*{\dosplits}{\ifnum\@k>0 \noalign{\hfil}\splitoff
2612 \global\advance\@k-1\cr\dosplits\fi}
2613
2614 \newcommand*{\splitoff}{\dimen0=\ht0
2615 \divide\dimen0 by\@k \advance\dimen0 by\@h
2616 \setbox2 \vsplit0 to \dimen0
2617 \unvbox2 }
2618
2619 %

```

### Three columns

```

\Xarrangement@threecol \newcommand*{\Xarrangement@threecol}[1]{%
2621 \csgdef{series@display#1}{\threecol}
2622 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
2623 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2624 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2625 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2626 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2627 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2628 \threecolfootsetup{#1}
2629 %

```

The additional setup for minipages.

```

2630 \ifnoledgroup@else
2631 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2632 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
2633 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2634 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2635 \mpthreecolfootsetup{#1}
2636 \fi
2637 }
2638
2639 %

```

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.

```
2640 \newcommand*{\threecolfootsetup}[1]{%
2641   \count\csname #1footins\endcsname 333
2642   \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2643   \multiply\dimen\csname #1footins\endcsname \thr@@}
2644   %
```

`\mpthreecolfootsetup` The setup for minipages.

```
2645 \newcommand*{\mpthreecolfootsetup}[1]{%
2646   \count\csname mp#1footins\endcsname 333
2647   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
2648   %
2649   %
```

`\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 `\hsize` to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal `\hsize` 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).

```
2650 \notbool{parapparatus@}{\newcommand*}{\newcommand*}{\threecolvfootnote}[2]{%
2651   \insert\csname #1footins\endcsname\bgroup%
2652   \hsize=\expandafter\dimexpr\csuse{Xhsize@#1}\relax%
2653   \noindent\csuse{Xhooknote@#1}%
2654   \csuse{Xnotefontsize@#1}%
2655   \footplitskips%
2656   \csname #1footfmt\endcsname #2{#1}\egroup}
2657   %
```

`\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.

```
2658 \notbool{parapparatus@}{\newcommand*}{\newcommand*}{\threecolfootfmt}[4]{%
2659   \normal@pars%
2660   \hsize \csuse{Xsizethreecol@#4}%
2661   \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2662   \tolerance=5000%
2663   \hangindent=\csuse{Xhangindent@#4}%
```



```

2664 \par%
2665 \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2666 \@tempdima=\parindent%
2667 \csuse{Xcolalign@#4}%
2668 \parindent=\@tempdima%
2669 \strut{\printlinefootnote{#1}{#4}}%
2670 {\nottoggle{Xlemmadisablefontselection@#4}}%
2671   {\select@lemmafont#1|{\csuse{Xlemmafont@#4}#2}}%
2672   {\csuse{Xlemmafont@#4}#2}}%
2673 }%
2674 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2675   {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2676   {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2677 }}%
2678 #3\strut\par\allowbreak}
2679 %

```

**\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.

```

2680 \newcommand*{\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2681 \noindent\csuse{Xtxtbeforenotes@#1}\csuse{Xbhookgroup@#1}}\par%
2682 \splittopskip=\ht\strutbox
2683 \expandafter
2684 \Xrigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
2685 %

```

**\mpthreecolfootgroup** The setup for minipages.

```

2686 \newcommand*{\mpthreecolfootgroup}[1]{\%
2687 \vskip\skip\@nameuse{mp#1footins}
2688 \ifl@dpairing\ifparledgroup%
2689   \leavevmode\marks\parledgroup@{begin}%
2690   \marks\parledgroup@series{#1}%
2691   \marks\parledgroup@type{Xfootnote}%
2692 \fi\fi\normalcolor
2693 \ifparledgroup%
2694   \ifl@dpairing%
2695   \else%
2696     \setXnoteswidthliketwocolumns@{#1}%

```

```

2697 \setXnotespositionliketwocolumns@{#1}%
2698 \print@Xfootnoterule{#1}%
2699 \fi%
2700 \else%
2701 \setXnoteswidthliketwocolumns@{#1}%
2702 \setXnotespositionliketwocolumns@{#1}%
2703 \print@Xfootnoterule{#1}%
2704 \fi%
2705 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2706 \csuse{Xbhookgroup@#1}\par%
2707 \splittopskip=\ht\strutbox
2708 \expandafter
2709 \Xrigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2710
2711 %

```

## Two columns

```

\Xarrangement@twocol12 \newcommand*{\Xarrangement@twocol}[1]{%
2713 \csgdef{series@display#1}{twocol}
2714 \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2715 \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
2716 \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2717 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2718 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2719 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2720 \twocolfootsetup{#1}
2721 %

```

The additional setup for minipages.

```

2722 \ifnoledgroup@else
2723 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2724 \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2725 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2726 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2727 \mptwocolfootsetup{#1}
2728 \fi
2729 }
2730
2731 %

```

`\twocolfootsetup` Here is a series of macros which are very similar to their three-column counterparts. In this case, each note is assumed to contribute only a half a line of text. And the notes are set in columns giving a gap between them of one tenth of the `\hsz`.

```

\Xarrangement@twocol12 \newcommand*{\twocolfootsetup}[1]{%
2732 \count\csname #1footins\endcsname 500
2733 \csxdef{default@#1footins}{500}%
2734

```

```

2735 \multiply\dimen\csname #1footins\endcsname \tw@}
2736 %

2737 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnote}[2]{%
2738 \insert\csname #1footins\endcsname\bgroup%
2739 \hsize=\expandafter\dimexpr\csuse{Xhsize@#1}\relax%
2740 \noindent\csuse{Xbhooknote@#1}%
2741 \csuse{Xnotefontsize@#1}%
2742 \footsplitskips%
2743 \csname #1footfmt\endcsname #2{#1}\egroup}
2744 %

2745 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmt}[4]{% 4th
arg is optional, for backward compatibility
2746 \normal@pars%
2747 \hsize \csuse{Xhsizetwocol@#4}%
2748 \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2749 \tolerance=5000%
2750 \hangindent=\csuse{Xhangindent@#4}%
2751 \par%
2752 \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2753 \@tempdima=\parindent%
2754 \csuse{Xcolalign@#4}%
2755 \parindent=\@tempdima%
2756 \strut{\printlinefootnote{#1}{#4}}%
2757 {\nottoggle{Xlemmadisablefontselection@#4}%
2758 {\select@lemmafont#1|{\csuse{Xlemmafont@#4}#2}}%
2759 {\csuse{Xlemmafont@#4}#2}}%
2760 }%
2761 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
Xlemmaseparator@#4}%
2762 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2763 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2764 }}%
2765 #3\strut\par\allowbreak}
2766 %

2767 \newcommand*{\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2768 \noindent\csuse{Xtxtbeforenotes@#1}\csuse{Xbhookgroup@#1}}\par%
2769 \splittopskip=\ht\strutbox
2770 \expandafter
2771 \Xrigidbalance\csname #1footins\endcsname \tw@ \splittopskip}
2772 %
2773 %

```

`\mptwocolfootsetup` The versions for minipages.

```

\mptwocolfootgroup
2774 \newcommand*{\mptwocolfootsetup}[1]{%
2775 \count\csname mp#1footins\endcsname 500

```

```

2776 \multiply\dimen\csname mp#1footins\endcsname \tw@}
2777 %

2778 \newcommand*{\mptwocolfootgroup}[1]{\{
2779 \vskip\skip\@nameuse{mp#1footins}
2780 \ifl@dpairing\ifparledgroup%
2781 \leavevmode\marks\parledgroup@{begin}%
2782 \marks\parledgroup@series{#1}%
2783 \marks\parledgroup@type{Xfootnote}%
2784 \fi\fi\normalcolor
2785 \ifparledgroup%
2786 \ifl@dpairing%
2787 \else%
2788 \setXnoteswidthliketwocolumns@{#1}%
2789 \setXnotespositionliketwocolumns@{#1}%
2790 \print@Xfootnoterule{#1}%
2791 \fi%
2792 \else%
2793 \setXnoteswidthliketwocolumns@{#1}%
2794 \setXnotespositionliketwocolumns@{#1}%
2795 \print@Xfootnoterule{#1}%
2796 \fi%
2797 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2798 \csuse{Xbhookgroup@#1}\par%
2799 \splittopskip=\ht\strutbox
2800 \expandafter
2801 \Xrigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2802
2803 %

```

## 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.

`\fullstop` 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.

```

2804 \def\endashchar{\textnormal{--}}
2805
2806 \newcommand*\fullstop{\textnormal{.}}
2807 \def\Xsublinesep@side{\fullstop}
2808
2809 \newcommand*\rbracket{\textnormal{%
2810   \csuse{text}\csuse{footnote@lang}}{%
2811     \ifluatex%
2812       \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]{\thinspace
2813         \else%
2814         \thinspace}%
2815         \fi}%
2816     }%
2817   }
2818
2819 %

```

### XII.7.2 Pstart number in footnote

`\printpstart` The `\printpstart` macro prints the pstart number for a note.

```

2820 \newcommand{\printpstart}[0]{%
2821   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
2822     l@dprintingcolumns}}{%
2823     \ifledRcol%
2824     \thepstartR%
2825     \else%
2826     \thepstartL%
2827     \fi%
2828   }{%
2829     \thepstart%
2830   }%
2831 %

```

### 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`.

```

2832 \newcommand{\printlinefootnote}[2]{%
2833   \l@d@p@rsefootspec#1|%
2834   \iftoggle{Xnumberonlyfirstintwolines@#2}{%

```

```

2835 \edef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub - \
l@dparsedendline - \l@dparsedendsub}%
2836 }%
2837 {%
2838 \edef\lineinfo@{\l@dparsedstartline - \l@dparsedstartsub}%
2839 }%
2840 \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2841 \hspace{\csuse{Xinplaceofnumber@#2}}%
2842 }%
2843 {%
2844 {%
2845 \iftoggle{Xnonumber@#2}%Try if the line number must printed (by
default, yes)
2846 {%
2847 \hspace{\csuse{Xinplaceofnumber@#2}}%
2848 }%
2849 {%
2850 {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2851 {%
2852 \ifcsdef{prevline#2}%
2853 {%Be sure the \prevline exists.
2854 \ifcsequal{prevline#2}{\lineinfo@}%Try it
2855 {%
2856 \ifcsequal{Xsymlinenum@#2}%
2857 {%
2858 \hspace{\csuse{Xinplaceofnumber@#2}}%
2859 }%
2860 {\printsymlinefootnotearea{#2}}%
2861 }%
2862 {%
2863 \printlinefootnotearea{#1}{#2}%
2864 }%
2865 }%
2866 {%
2867 \printlinefootnotearea{#1}{#2}%
2868 }%
2869 }%
2870 {%
2871 \printlinefootnotearea{#1}{#2}%
2872 }%
2873 \csxdef{prevline#2}{\lineinfo@}%
2874 }%
2875 }%
2876 }%
2877 }%
2878 }
2879 %

```

**\printsymlinefootnotearea** This macro prints the space before the line symbol, changes the font, when prints the line symbol and the space after it.

```

2880 \newcommand{\printsymlinefootnotearea}[1]{%
2881   \hspace{\csuse{Xbeforesymlinenum@#1}}%
2882   \csuse{Xnotenumfont@#1}%
2883   \ifdimequal{\csuse{Xboxsymlinenum@#1}}{\z@}%
2884     {\csuse{Xsymlinenum@#1}}%
2885     {\hbox to \csuse{Xboxsymlinenum@#1}%
2886       {\csuse{Xsymlinenum@#1}\hfill}%
2887     }%
2888   \hspace{\csuse{Xaftersymlinenum@#1}}%
2889 }%
2890 %

```

**\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.)

```

2891 \newcommand{\printlinefootnotearea}[2]{%
2892   \printXbeforenumber{#2}%
2893   \csuse{Xnotenumfont@#2}%
2894   \boxfootnotenumbers{#1}{#2}%
2895   \printXafternumber{#2}%
2896 }%
2897 %

```

**\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.

```

2898 \newcommand{\boxfootnotenumbers}[2]{%
2899   \ifdimequal{\csuse{Xboxlinenum@#2}}{\Opt}{%
2900     \printlinefootnotenumbers{#1}{#2}%
2901   }%
2902   {%
2903     \hbox to \csuse{Xboxlinenum@#2}%
2904     {%
2905       \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2906       \printlinefootnotenumbers{#1}{#2}%
2907       \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2908     }%
2909   }%
2910 }%
2911 %

```

**\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.

```

2912 \newcommand{\printlinefootnotenumbers}[2]{%
2913   \xdef\@currentseries{#2}%
2914   \ifboolexpr{%
2915     (togl{Xpstart@#2} and bool{numberpstart})%
2916     or togl{Xpstarteverytime@#2}}%
2917   {\printpstart}{}%
2918   \iftoggle{Xstanza@#2}{%
2919     \ifnumberstanza%
2920     \printstanza%
2921     \csuse{Xstanzaseparator@#2}%
2922     \fi%
2923   }{%
2924     \iftoggle{Xonlypstart@#2}{}%
2925     \csuse{Xtxtbeforenumber@#2}%
2926     \printlines#1||\ifledRcol@{\@Rlineflag}\fi}%
2927   }%
2928   %

```

**\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.)

```

2929 \newcommand{\printXbeforenumber}[1]{%
2930   \hspace{\csuse{Xbeforenumber@#1}}%
2931 }%
2932 %

```

**\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

```

2933 \newcommand{\printXafternumber}[1]{%
2934   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%
2935   \hspace{\csuse{Xafternumber@#1}}%
2936 }%
2937 %

```

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. 86: 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.

Maieul 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@pnum38 \newif\ifl@d@pnum
\ifl@d@ssub39 \newif\ifl@d@ssub
\ifl@d@elin40 \newif\ifl@d@elin
\ifl@d@esl41 \newif\ifl@d@esl
\ifl@d@dash42 \newif\ifl@d@dash
\ifl@d@Xtwolines43 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines44 \newif\ifl@d@Xmorethantwolines%
45 %

\l@dp@rsefootsspec \l@dp@rsefootsspec parses lines specification and defines macros which hold the nu-
\l@dparsedstartpage \l@dparsedstartpage{#1}%
\l@dparsedstartline \printlines #1 | #2 | #3 | #4 | #5 | #6 | #7
\l@dparsedstartsub \printlines start-page | line | subline | end-page | line | subline | fontflag
\l@dparsedendpage \def\l@dp@rsefootsspec#1|#2|#3|#4|#5|#6|#7|{%
46 \gdef\l@dparsedstartpage{#1}%
47 \gdef\l@dparsedstartline{#2}%
48 \gdef\l@dparsedstartsub{#3}%
49 \gdef\l@dparsedendpage{#4}%
50 \gdef\l@dparsedendline{#5}%
51 \gdef\l@dparsedendsub{#6}%
52 }
53 %
54 %

```

Initialise the several number value macros.

```

55 \def\l@dparsedstartpage{0}%
56 \def\l@dparsedstartline{0}%
57 \def\l@dparsedstartsub{0}%
58 \def\l@dparsedendpage{0}%
59 \def\l@dparsedendline{0}%
60 \def\l@dparsedendsub{0}%
61 %
62 %

```

`\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`.

```

63 \newcommand*\setprintlines[6]{%
64 \l@d@pnumfalse \l@d@dashfalse
65 %

```

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

```
2966 \ifbypage@
2967   \ifnum#4=#1 \else
2968     \l@d@pnumtrue
2969     \l@d@dashtrue
2970   \fi
2971 \fi
2972 %
```

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.

```
2973 \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
2974 \ifnum#2=#5 \else
2975   \l@d@elintrue
2976   \l@d@dashtrue
2977 \fi
2978 %
```

We print the starting sub-line if it is nonzero.

```
2979 \l@d@ssubfalse
2980 \ifnum#3=0 \else
2981   \l@d@ssubtrue
2982 \fi
2983 %
```

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.

```
2984 \l@d@eslfalse
2985 \ifnum#6=0 \else
2986   \ifnum#6=#3
2987     \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
2988   \else
2989     \l@d@esltrue
2990     \l@d@dashtrue
2991   \fi
2992 \fi%
2993 %
```

However, if the `\Xtwolines` is set for the current series, we do not print the last line number.

```
2994 \ifl@d@dash%
2995   \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xtwolines@}\
2996   \currentseries}}}%
2997   {}%
2998   {}%
2999   \setistwofollowinglines{#1}{#2}{#4}{#5}%
3000   \ifboolexpr{%
```

```

3000      (%)
3001      togl {Xtwolinesbutnotmore@\@currentseries}%
3002      and not%
3003      (%)
3004      bool {istwofollowinglines@}%
3005      )%
3006  )%
3007  or%
3008  (%)
3009      (not test{\ifnumequal{#1}{#4}})%
3010      and togl{Xtwolinesonlyinsamepage@\@currentseries}%
3011      )%
3012  }%
3013  {}%
3014  {%
3015      \l@d@dashfalse%
3016      \l@d@Xtwolinestrue%
3017      \l@d@elinfalse%
3018      \l@d@eslfalse%
3019      \ifcempty{Xmorethantwolines@\@currentseries}%
3020      {%
3021          {\ifistwofollowinglines\else%
3022              \l@d@Xmorethantwolinestrue%
3023              \fi%
3024          }%
3025      }%
3026  }%
3027  \fi%
3028  %
3029  }%
3030  %

```

End of \setprintlines.

**\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.

```

3031 \newif\ifistwofollowinglines@%
3032 \newcommand{\setistwofollowinglines}[4]{%
3033   \ifcsdef{lastlinenumberon@#1}%
3034     {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
3035     {\numdef{\tmp}{0}}%
3036   \istwofollowinglines@false%
3037   \ifnumequal{#4-#2}{1}%
3038     {\istwofollowinglines@true}%
3039     {\ifbypage@%
3040       \ifnumequal{#3-#1}{1}%
3041       {%
3042         \ifnumequal{#2}{\tmp}%
3043         {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
3044         {}}%
3045       }%
3046       {}%
3047     \fi%
3048   }%
3049 }%
3050 %

```

`\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

```

3051 \def\printlines#1|#2|#3|#4|#5|#6|#7|#8|{%
3052   \begingroup%
3053   %

```

If we use Lua<sub>T</sub><sub>E</sub>X, ensure we use good text's direction.

```

3054   \ifluatex%
3055     \edef\@tmp{\the\textdir}%
3056     \ifdefstring{\@tmp}{TLT}{\textdir TLT}%Test in order to prevent
3057     spurious space (bug #397)
3058   \fi%
3059   %

```

Decide which part of line number components we will print.

```

3059   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
3060   %

```

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.

```

3061 \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
3062   {\bgroup}%
3063   {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
hfill}%
3064 \ifl@d@pnum%
3065   \wrap@edcrossref{\@this@crossref@start}{#1}%
3066   \csuse{Xsublinesep@\@currentseries}%
3067 \fi%
3068 \wrap@edcrossref{\@this@crossref@start}{%
3069   \linenumrep{#2}%
3070   \iftoggle{Xlineflag@\@currentseries}{#8}{}}%
3071 }%
3072 \ifl@d@ssub%
3073   \csuse{Xsublinesep@\@currentseries}%
3074   \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
3075 \fi
3076 \egroup%
3077 %

```

Then print the dash + end line number, or the range symbol.

```

3078 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3079   {\bgroup}%
3080   {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
3081 \ifl@d@Xtwolines%
3082   \ifl@d@Xmorethantwolines%
3083     \csuse{Xmorethantwolines@\@currentseries}%
3084   \else%
3085     \csuse{Xtwolines@\@currentseries}%
3086   \fi%
3087 \else%
3088   \ifl@d@dash%
3089     \ifdefined\linerrangesep%
3090       \linerrangesep%
3091     \else%
3092       \csuse{Xlinerrangeseparator@\@currentseries}%
3093     \fi%
3094   \fi%
3095   \ifl@d@pnum%
3096     \wrap@edcrossref{\@this@crossref@end}{#4}%
3097     \csuse{Xsublinesep@\@currentseries}%
3098   \fi%
3099   \ifl@d@elin%
3100     \wrap@edcrossref{\@this@crossref@end}{%
3101       \linenumrep{#5}%
3102       \iftoggle{Xlineflag@\@currentseries}{#8}{}}%
3103   }%
3104 \fi%
3105 \ifl@d@esl%
3106   \ifl@d@elin%
3107     \csuse{Xsublinesep@\@currentseries}%

```

```

3108     \fi%
3109     \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}}%
3110     \fi%
3111     \fi%
3112     \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3113     {}%
3114     {\hfill}%Prevent underfull hbox
3115     \egroup%
3116     \endgroup%
3117 }%
3118 %

```

## XIII Familiar footnotes

### XIII.1 Adjacent footnotes

The original edmac provided users with five series of critical footnotes (`\Afootnote` `\Bfootnote` `\Cfootnote` `\Dfootnote` `\Efootnote`), and  $\text{\LaTeX}$  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` 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`.

```

3119 \providecommand*\multiplefootnotemarker{3sp}
3120 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
3121
3122 %

```

`\m@mmf@prepare` A pair of self-cancelling kerns. This may have been defined in the `memoir` class.

```

3123 \providecommand*\m@mmf@prepare{%
3124     \kern-\multiplefootnotemarker
3125     \kern\multiplefootnotemarker\relax}
3126 %

```

`\m@mmf@check` This may have been defined in the `memoir` class. If it recognises the last kern as `\multiplefootnotemarker` it typesets `\multfootsep`.

```

3127 \providecommand*\m@mmf@check{%
3128     \ifdim\lastkern=\multiplefootnotemarker\relax
3129     \edef\x@sf{\the\spacefactor}%

```

```

3130 \unkern
3131 \multfootsep
3132 \spacefactor\@xsf\relax
3133 \fi}
3134
3135 %

```

We have to modify `\@footnotetext` and `\@footnotemark`. However, if `memoir` is used the modifications have already been made.

```

3136 \@ifclassloaded{memoir}{}%
3137 %

```

`\@footnotetext` Add `\m@mmf@prepare` at the end of `\@footnotetext`.

```

3138 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3139 %

```

`\@footnotemark` Modify `\@footnotemark` to cater for adjacent `\footnotes`.

```

3140
3141 \patchcmd{\@footnotemark}
3142 {\nobreak}
3143 {\m@mmf@check
3144 \nobreak
3145 }
3146 {}{}
3147 \patchcmd{\@footnotemark}
3148 {\@makefnmark}
3149 {\@makefnmark
3150 \m@mmf@prepare
3151 }
3152 {}{}
3153 %

```

Finished the modifications for the non-memoir case.

```

3154 }
3155
3156 %

```

## 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 with its `\@footnotetext`, using different forms for when in numbered or regular text.

```

3157 \pretocmd{\@footnotetext}{%
3158   \ifnumberedpar@
3159   \edtext{}\{\l@dbfnote{#1}}}%
3160 \else
3161   {}{}
3162 \apptocmd{\@footnotetext}{\fi}{}{}%
3163 %

```

`\l@dbfnote`    `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original  
`\vl@dbfnote`    `\@footnotetext`. We also patch `\footnote` in order to get the correct footnote  
`\vl@dbfnote`    numbers when typesetting parallel texts. This is moved into a `\get@fnmark` command.  
`\footnote`  
`\get@fnmark`  
`\get@thisfootnote`

```

3164 \get@thisfootnote
3165 \patchcmd%
3166   {\footnote}%
3167   {\stepcounter\@mpfn}%
3168   {%
3169   \ifl@dpairing%
3170   \global\advance\footnote@reading by \@ne%
3171   \get@thisfootnote%
3172   \get@fnmark{\thisfootnote}%
3173   \ifcsdef{footnotereading\the\footnote@reading=typeset}%
3174   {\setcounter{\@mpfn}{\csuse{footnotereading\the\footnote@reading=
typeset}}}%
3175   {\setcounter{\@mpfn}{\footnote@reading}}}%
3176   \else%
3177   \stepcounter\@mpfn%
3178   \fi%
3179   }%
3180   {}
3181   {}
3182
3183 \newcommand{\get@thisfootnote}{%
3184   \ifl@dpairing
3185   \protected@xdef\thisfootnote{\the\footnote@reading}%
3186   \else%
3187   \protected@xdef\thisfootnote{\thefootnote}%
3188   \fi%
3189 }%
3190
3191 \newcommand{\l@dbfnote}[1]{%
3192   \get@thisfootnote%
3193   \gdef\@tag{#1\relax}%
3194   \ifledRcol%
3195   \xright@appenditem{%
3196     \ifdefined\Hy@footnote@currentHref%
3197     \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3198     \fi%

```



```

3199 \noexpand\vl@dbfnote{\expandonce\@tag}{\thisfootnote}%
3200 }%
3201 \to\inserts@listR
3202 \global\advance\insert@countR \@ne%
3203 \else%
3204 \xright@appenditem{%
3205 \ifdefined\Hy@footnote@currentHref%
3206 \noexpand\def\noexpand\Hy@footnote@currentHref{\
Hy@footnote@currentHref}%
3207 \fi%
3208 \noexpand\vl@dbfnote{\expandonce\@tag}{\thisfootnote}%
3209 }%
3210 \to\inserts@list
3211 \global\advance\insert@count \@ne%
3212 \fi
3213 \ignorespaces%
3214 }%
3215
3216 \newcommand{\get@fnmark}[1]{%
3217 \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}%
3218 {%
3219 \stepcounter{footnote@typeset}%
3220 \setcounter{footnote}{\c@footnote@typeset}%
3221 \immediate\write\@mainaux{%
3222 \global\csdef{footnotereading#1=typeset}{\the\c@footnote@typeset}
%
3223 }%
3224 \def\@thefnmark{\thefootnote}%
3225 }%
3226 {%
3227 \@namedef{\@thefnmark}{#1}%
3228 }%
3229 }%
3230
3231 \newcommand{\vl@dbfnote}[2]{%
3232 \get@fnmark{#2}%
3233 \@footnotetext{#1}%
3234 }%
3235 %

```

### 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`  
3236 `\newcommand*{\prebodyfootmark}{%`

```

3237 \leavevmode
3238 \ifhmode
3239   \edef\x@sf{\the\spacefactor}%
3240   \m@mmf@check
3241   \nobreak
3242   \fi}
3243 \newcommand{\postbodyfootmark}{%
3244   \m@mmf@prepare
3245   \ifhmode\spacefactor\x@sf\fi\relax}
3246
3247 %

```

## 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.

```

3248 \newcommandx{\arrangementX}[2][1,usedefault]{%
3249   \def\do##1{%
3250     \csname arrangementX@#2\endcsname{##1}%
3251   }%
3252   \ifstrempy{#1}%
3253     {%
3254       \dolistloop{\@series}%
3255     }%
3256     {
3257       \docsvlist{#1}%
3258     }%
3259   }%
3260 %

```

### 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.

```

3261 \newcommand*{\normal@footnotemarkX}[1]{%
3262   \prebodyfootmark
3263   \wrapped@bodyfootmarkX{#1}%
3264   \postbodyfootmark}
3265
3266 %

```

**\normalbodyfootmarkX** The `\normalbodyfootmarkX{series}` really typesets the in-text marker. The style is the normal superscript.

```

3267 \newcommand*{\normalbodyfootmarkX}[1]{%
3268   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}

```

3269 %

**\normalvfootnoteX** `\normalvfootnoteX{<series>}{<text>}` does the `\insert` for the `<series>` and calls the series' `\footfmt...` to format the `<text>`.

```

3270 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalvfootnoteX}[2]{%
3271   \insert\@nameuse{footins#1}\bgroup
3272   \hsize=\expandafter\dimexpr\csuse{hsizeX@#1}\relax%
3273   \noindent\csuse{bhooknoteX@#1}%
3274   \csuse{notefontsizeX@#1}%
3275   \footssplitskips
3276   \ifl@dpairing\ifl@dpaging\else%
3277     \setnoteswidthliketwocolumnsX@{#1}%
3278   \fi\fi%
3279   \setnotesXpositionliketwocolumns@{#1}%
3280   \spaceskip=\z@skip \xspaceskip=\z@skip
3281   \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3282 %
3283 %

```

**\mpnormalvfootnoteX** The minipage version.

```

3284 \newcommand*{\mpnormalvfootnoteX}[2]{%
3285   \get@thisfootnoteX{#1}%
3286   \get@fnmarkX{#1}{\thisfootnote}%
3287   \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3288   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3289     \unvbox\@nameuse{mpfootins#1}
3290     \noindent\csuse{bhooknoteX@#1}%
3291     \csuse{notefontsizeX@#1}%
3292     \hsize\columnwidth
3293     \@parboxrestore
3294     \color@begingroup
3295     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3296 %
3297 %

```

**\normalfootfmtX** `\normalfootfmtX{<series>}{<text>}` typesets the footnote text, prepended by the marker.

```

3298 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalfootfmtX}[2]{%
3299   \ifluatex%
3300     \texdir\footnote@luatextextdir%
3301     \pardir\footnote@luatexpardir%
3302     \par%
3303   \fi%
3304   \protected@edef\@currentlabel{%
3305     \@nameuse{@thefnmark#1}%
3306   }%
3307   \ledsetnormalparstuffX{#1}%
3308   \hangindent=\csuse{hangindentX@#1}%

```

```

3309 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3310 {{\csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}}\strut%
3311 #2\strut\par}}
3312
3313 %

```

**\normalfootfootmarkX** `\normalfootfootmarkX{<series>}` is called by `\normalfootfmtX` to typeset the footnote marker in the footer before the footnote text.

```

3314 \newcommand*{\normalfootfootmarkX}[1]{%
3315 \textsuperscript{\@nameuse{@thefnmark#1}}}
3316
3317 %

```

**\normalfootstartX** `\normalfootstartX{<series>}` is the `<series>` footnote starting macro used in the output routine.

```

3318 \newcommand*{\normalfootstartX}[1]{%
3319 \ifdimequal{Opt}{\prenotesX@}{}%
3320 {%
3321 \iftoggle{prenotesX@}{%
3322 \togglefalse{prenotesX@}%
3323 \skip\csname footins#1\endcsname=%
3324 \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3325 }%
3326 {}%
3327 }%
3328 \vskip\skip\csname footins#1\endcsname%
3329 \leftskip=\z@
3330 \rightskip=\z@
3331 \ifl@dpairing\else%
3332 \hsize=\old@hsize%
3333 \fi%
3334 \setnoteswidthliketwocolumnsX@{#1}%
3335 \setnotesXpositionliketwocolumns@{#1}%
3336 \print@footnoteXrule{#1}%
3337 }%
3338
3339 %

```

**\normalfootnoteruleX** The rule drawn before the footnote series group.

```

3340 \let\normalfootnoteruleX=\footnoterule
3341
3342 %

```

**\normalfootgroupX** `\normalfootgroupX{<series>}` sends the contents of the `<series>` insert box to the output page without alteration.

```

3343 \newcommand*{\normalfootgroupX}[1]{%
3344   \csuse{bhookgroupX@#1}%
3345   \unvbox\@nameuse{footins#1}%
3346   \hsize=\old@hsize%
3347 }%
3348
3349 %

```

**\mpnormalfootgroupX** The minipage version.

```

3350 \newcommand*{\mpnormalfootgroupX}[1]{%
3351   \vskip\skip\@nameuse{mpfootins#1}
3352   \ifl@dpairing\ifparledgroup%
3353     \leavevmode\marks\parledgroup@{begin}%
3354     \marks\parledgroup@series{#1}%
3355     \marks\parledgroup@type{footnoteX}%
3356   \fi\fi\normalcolor
3357   \ifparledgroup%
3358     \ifl@dpairing%
3359     \else%
3360       \setnoteswidthliketwocolumnsX@{#1}%
3361       \setnotesXpositionliketwocolumns@{#1}%
3362       \print@footnoteXrule{#1}%
3363     \fi%
3364   \else%
3365     \setnoteswidthliketwocolumnsX@{#1}%
3366     \setnotesXpositionliketwocolumns@{#1}%
3367     \print@footnoteXrule{#1}%
3368   \fi%
3369   \csuse{bhookgroupX@#1}%
3370   \unvbox\@nameuse{mpfootins#1}}
3371
3372 %

```

**\normalbfnoteX**<sub>73</sub>

```

3374 \newcommand{\normalbfnoteX}[2]{%
3375   \get@thisfootnoteX{#1}%
3376   \ifledRcol%
3377     \ifluatex
3378       \footnotelang@lua[R]%
3379     \fi
3380     \@ifundefined{xpg@main@language}%if polyglossia
3381     {}%
3382     {\footnotelang@poly[R]}%
3383     \xright@appenditem{%
3384       \noexpand\led@set@index@fornote{#1}%
3385       \unexpanded{\def\this@footnoteX@reading}{\the\cename footnote#1
@reading\endcsname}%
3386       \noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}%

```

```

3387 \noexpand\led@reinit@index@fornote%
3388 }%
3389 \to\inserts@listR
3390 \global\advance\insert@countR \@ne%
3391 \else%
3392 \ifluatex
3393 \footnotelang@lua%
3394 \fi
3395 \ifundefined{xpg@main@language}%if polyglossia
3396 {}%
3397 {\footnotelang@poly}%
3398 \xright@appenditem{%
3399 \noexpand\led@set@index@fornote{#1}%
3400 \unexpanded{\def\this@footnoteX@reading}{\the\csname footnote#1
@reading\endcsname}%
3401 \noexpand\vbfnoteX{#1}{#2}{\expandonce\thisfootnote}%
3402 \noexpand\led@reinit@index@fornote%
3403 }%
3404 \to\inserts@list
3405 \global\advance\insert@count \@ne%
3406 \fi
3407 \ignorespaces}
3408
3409 %

```

**\get@thisfootnoteX** The macro `\get@thisfootnote` command just saves the footnote number in the `\thisfootnote` macro, depending on the use of pairing environments.

```

3410 \newcommand{\get@thisfootnoteX}[1]{%
3411 \ifl@dpairing%
3412 \protected@xdef\thisfootnote{\the\csname footnote#1@reading\endcsname
}%
3413 \else%
3414 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3415 \fi%
3416 }%
3417 %

```

**\vbfnoteX** This command calls the correct footnote-inserting commands.

```

3418 \newcommand{\vbfnoteX}[3]{%
3419 \get@fnmarkX{#1}{#3}%
3420 \@nameuse{regvfootnote#1}{#1}{#2}%
3421 }%
3422
3423 %

```

**\get@fnmarkX** This command gets the correct footnote number when typesetting parallel texts.

```

3424 \newcommand{\get@fnmarkX}[2]{%
3425   \ifboolexpr{bool{!@dpairing} or bool{!@dprintingpages} or bool{
1@dprintingcolumns}}{%
3426     {%
3427       \stepcounter{footnote#1@typeset}%
3428       \setcounter{footnote#1}{\value{footnote#1@typeset}}%
3429       \@namedef{@thefnmark#1}{\csuse{thefootnote#1}}%
3430       \immediate\write\@mainaux{%
3431         \global\csdef{footnote#1reading#2=typeset}{\the\csname c@footnote
#1@typeset\endcsname}%
3432       }%
3433     }%
3434     {%
3435       \@namedef{@thefnmark#1}{#2}%
3436     }%
3437   }
3438   %
3439   %

```

```

\vnunfootnoteX40 \newcommand{\vnunfootnoteX}[2]{%
3441   \ifnumberedpar@
3442     \edtext{}{\normalbfnoteX{#1}{#2}}%
3443   \else
3444     \def\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3445     \@nameuse{regvfootnote#1}{#1}{#2}%
3446   \fi}
3447
3448   %

```

`arrangementX@normal` `\arrangementX@normal{<series>}` initialises the settings for the `<series>` footnotes. This should always be called for each series.

```

3449 \newcommand*{\arrangementX@normal}[1]{%
3450   \csgdef{series@displayX#1}{normal}
3451   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3452   \expandafter\newcount\csname prevpage#1@num\endcsname
3453   \@namedef{footnotemark#1}{\normal@footnotemarkX{#1}}
3454   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3455   \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3456   \expandafter\let\csname vfootnote#1\endcsname=\vnunfootnoteX
3457   \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3458   \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3459   \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3460   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3461   \count\csname footins#1\endcsname=1000
3462   \csxdef{default@footins#1}{1000}%Use to have note only for one side
3463   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3464   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3465   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%

```

```

3466 %
Additions for minipages.
3467 \ifnoledgroup@else%
3468   \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3469   \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3470   \count\csname mpfootins#1\endcsname=1000
3471   \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}%
3472   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3473   \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3474 \fi
3475 }
3476
3477 %

```

### 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 3478 \newcommand*{\arrangementX@twocol}[1]{%
3479   \csgdef{series@displayX#1}{twocol}
3480   \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
3481   \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3482   \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3483   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3484   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3485   \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3486   \twocolfootsetupX{#1}
3487   \ifnoledgroup@else%
3488     \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3489     \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
3490     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3491     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
3492     \mptwocolfootsetupX{#1}
3493   \fi%
3494 }
3495
3496 %

```

```

\twocolfootsetupX \twocolfootsetupX{<series>}
\mptwocolfootsetupX
3497 \newcommand*{\twocolfootsetupX}[1]{%
3498   \count\csname footins#1\endcsname 500
3499   \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3500   \multiply\dimen\csname footins#1\endcsname by \tw@
3501 \newcommand*{\mptwocolfootsetupX}[1]{%
3502   \count\csname mpfootins#1\endcsname 500

```



```

3503 \multiply\dimen\csname mpfootins#1\endcsname by \tw@
3504
3505 %

```

**\twocolvfootnoteX** \twocolvfootnoteX{<series>}

```

3506 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
3507 \insert\csname footins#1\endcsname\bgroup%
3508 \hsize=\expandafter\dimexpr\csuse{hsizeX@#1}\relax%
3509 \noindent\csuse{bhooknoteX@#1}%
3510 \csuse{notefontsizeX@#1}%
3511 \footsplitskips%
3512 \spaceskip=\z@skip \xspaceskip=\z@skip%
3513 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3514
3515 %

```

**\twocolfootfmtX** \twocolfootfmtX{<series>}

```

3516 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3517 \protected@edef\@currentlabel{%
3518 \@nameuse{@thefnmark#1}%
3519 }%
3520 \normal@pars%
3521 \hangindent=\csuse{hangindentX@#1}%
3522 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3523 \hsize \csuse{hsizetwocolX@#1}%
3524 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3525 \tolerance=5000\relax%
3526 \par%
3527 \@tempdima=\parindent%
3528 \csuse{colalignX@#1}%
3529 \parindent=\@tempdima%
3530 {\csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
3531 #2\strut\par}\allowbreak}
3532
3533 %

```

**\twocolfootgroupX** \twocolfootgroupX{<series>}

**\mptwocolfootgroupX**

```

3534 \newcommand*{\twocolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3535 \splittopskip=\ht\strutbox
3536 \expandafter
3537 \rigidbalanceX\csname footins#1\endcsname \tw@ \splittopskip}}
3538
3539 \newcommand*{\mptwocolfootgroupX}[1]{%
3540 \vskip\skip\@nameuse{mpfootins#1}
3541 \ifl@dpairing\ifparledgroup%
3542 \leavevmode\marks\parledgroup@{begin}%

```

```

3543 \marks\parledgroup@series{#1}%
3544 \marks\parledgroup@type{footnoteX}%
3545 \fi\fi\normalcolor
3546 \ifparledgroup%
3547 \ifl@dpairing%
3548 \else%
3549 \setnoteswidthliketwocolumnsX@{#1}%
3550 \setnotesXpositionliketwocolumns@{#1}%
3551 \print@footnoteXrule{#1}%
3552 \fi%
3553 \else%
3554 \setnoteswidthliketwocolumnsX@{#1}%
3555 \setnotesXpositionliketwocolumns@{#1}%
3556 \print@footnoteXrule{#1}%
3557 \fi%
3558 \csuse{bhookgroupX@#1}%
3559 \splittopskip=\ht\strutbox
3560 \expandafter
3561 \rigidbalanceX\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3562
3563 %

```

#### 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 64 \newcommand*{\arrangementX@threecol}[1]{%
3565 \csgdef{series@displayX#1}{threecol}
3566 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3567 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3568 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3569 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3570 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3571 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3572 \threecolfootsetupX{#1}
3573 \ifnoledgroup@\else%
3574 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3575 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3576 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3577 \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3578 \mpthreecolfootsetupX{#1}
3579 \fi%
3580 }
3581
3582 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX

```

```

3583 \newcommand*{\threecolfootsetupX}[1]{%
3584   \count\csname footins#1\endcsname 333
3585   \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3586   \multiply\dimen\csname footins#1\endcsname by \thr@@
3587 \newcommand*{\mpthreecolfootsetupX}[1]{%
3588   \count\csname mpfootins#1\endcsname 333
3589   \multiply\dimen\csname mpfootins#1\endcsname by \thr@@
3590
3591   %

```

**\threecolvfootnoteX** **\threecolvfootnoteX{<series>}{<text>}**

```

3592 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{
%
3593   \insert\csname footins#1\endcsname\bgroup%
3594     \hsize=\expandafter\dimexpr\csuse{hsizeX@#1}\relax%
3595     \noindent\csuse{bhooknoteX@#1}%
3596     \csuse{notefontsizeX@#1}
3597     \footssplitsskips%
3598     \@nameuse{footfmt#1}{#1}{#2}\egroup}
3599
3600   %

```

**\threecolfootfmtX** **\threecolfootfmtX{<series>}**

```

3601 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3602   \protected@edef\@currentlabel{%
3603     \@nameuse{@thefnmark#1}%
3604   }%
3605   \hangindent=\csuse{hangindentX@#1}%
3606   \everypar{\hangindent=\csuse{hangindentX@#1}}%
3607   \normal@pars%
3608   \hsize \csuse{hsizethreecolX@#1}%
3609   \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3610   \tolerance=5000\relax%
3611   \par%
3612   \@tempdima=\parindent%
3613   \csuse{colalignX@#1}%
3614   \parindent=\@tempdima%
3615   {\csuse{notenumfontX@#1}\wrapped@footfootmarkX{#1}\strut%
3616     #2\strut\par}\allowbreak}
3617
3618   %

```

**\threecolfootgroupX** **\threecolfootgroupX{<series>}**  
**\mpthreecolfootgroupX**

```

3619 \newcommand*{\threecolfootgroupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3620   \splittopskip=\ht\strutbox

```

```

3621 \expandafter
3622 \rigidbalanceX\csname footins#1\endcsname \thr@@ \splittopskip}}
3623
3624 \newcommand*{\mpthreecolfootgroupX}[1]{%
3625 \vskip\skip\@nameuse{mpfootins#1}
3626 \ifl@dpairing\ifparledgroup
3627 \leavevmode\marks\parledgroup@{begin}%
3628 \marks\parledgroup@series{#1}%
3629 \marks\parledgroup@type{footnoteX}%
3630 \fi\fi\normalcolor
3631 \ifparledgroup%
3632 \ifl@dpairing%
3633 \else%
3634 \setnoteswidthliketwocolumnsX@{#1}%
3635 \setnotesXpositionliketwocolumns@{#1}%
3636 \print@footnoteXrule{#1}%
3637 \fi%
3638 \else%
3639 \setnoteswidthliketwocolumnsX@{#1}%
3640 \setnotesXpositionliketwocolumns@{#1}%
3641 \print@footnoteXrule{#1}%
3642 \fi%
3643 \csuse{bhookgroupX@#1}%
3644 \splittopskip=\ht\strutbox
3645 \expandafter
3646 \rigidbalanceX\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3647
3648 %

```

### XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

`\arrangementX@threecol` `\footparagraphX{<series>}`

```

3649 \newcommand*{\arrangementX@paragraph}[1]{%
3650 \csgdef{series@displayX#1}{paragraph}%
3651 \expandafter\newcount\csname #1prevpage@num\endcsname
3652 \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3653 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
3654 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3655 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3656 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3657 \count\csname footins#1\endcsname=1000
3658 \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3659 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3660 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3661 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3662 \para@footsetupX{#1}

```

```

3663 \ifnoledgroup@ \else
3664   \expandafter \let \csname mpvfootnote#1 \endcsname = \mppara@vfootnoteX
3665   \expandafter \let \csname mpfootgroup#1 \endcsname = \mppara@footgroupX
3666   \count \csname mpfootins#1 \endcsname = 1000
3667   \dimen \csname mpfootins#1 \endcsname = \csuse{maxhnotesX@#1}
3668   \skip \csname mpfootins#1 \endcsname = \csuse{beforenotesX@#1}%
3669   \advance \skip \csname mpfootins#1 \endcsname by \csuse{afterruleX@#1}%
3670 \fi
3671 }
3672
3673 %

```

`\para@footsetupX` `\para@footsetupX{<series>}`

```

3674 \newcommand*{\para@footsetupX}[1]{\csuse{bhookgroupX@#1}\csuse{
notefontsizeX@#1}
3675   \setnoteswidthliketwocolumnsX@{#1}%
3676   \ifcempty{hsizeX@#1}%
3677     {}%
3678     {\columnwidth = \expandafter \dimexpr \csuse{hsizeX@#1} \relax}%
3679   \dimen0 = \baselineskip
3680   \multiply \dimen0 by 1024
3681   \divide \dimen0 by \columnwidth \multiply \dimen0 by \footfudgefiddle \relax
3682   %
3683   \expandafter
3684   \xdef \csname footfudgefactor#1 \endcsname {%
\expandafter \strip@pt \dimen0 }}
3685
3686 %

```

`\parafootstartX` `\parafootstartX{<series>}`

```

3687 \newcommand*{\parafootstartX}[1]{%
3688   \ifdimequal{0pt}{\prenotesX@}{}%
3689     {%
3690       \iftoggle{prenotesX@}{%
3691         \togglefalse{prenotesX@}%
3692         \skip \csname footins#1 \endcsname = %
3693         \dimexpr \csuse{prenotesX@} + \csuse{afterruleX@#1} \relax%
3694       }%
3695     }%
3696   }%
3697   \leftskip = \z@
3698   \rightskip = \z@
3699   \nottoggle{parindentX@#1}{\parindent = \z@}{}%
3700   \vskip \skip \@nameuse{footins#1}%
3701   \setnoteswidthliketwocolumnsX@{#1}%
3702   \setnotesXpositionliketwocolumns@{#1}%
3703   \print@footnotexrule{#1}%
3704 }

```

```

3705
3706 %

\para@vfootnoteX \para@vfootnoteX{<series>}{<text>}
\mppara@vfootnoteX
3707 \newcommand*{\para@vfootnoteX}[2]{%
3708 \insert\csname footins#1\endcsname%
3709 \bgroup
3710 \csuse{notefontsizeX@#1}
3711 \footsplitskips
3712 \setbox0=\vbox{\hsize=\maxdimen%
3713 \let\bidirTL@everypar\@empty%
3714 \noindent\csuse{bhooknoteX@#1}%
3715 \@nameuse{footfmt#1}{#1}{#2}}%
3716 \setbox0=\hbox{\unvXH{0}{#1}}%
3717 \dp0=\z@
3718 \ht0=\csname footfudgefactor#1\endcsname\wd0
3719 \box0
3720 \penalty0
3721 \egroup}
3722 \newcommand*{\mppara@vfootnoteX}[2]{%
3723 \get@thisfootnoteX{#1}%
3724 \get@fnmarkX{#1}{\thisfootnote}%
3725 \edef\this@footnoteX@reading{\the\csname footnote#1@reading\endcsname}%
3726 \global\setbox\@nameuse{mpfootins#1}\vbox{%
3727 \unvbox\@nameuse{mpfootins#1}
3728 \csuse{notefontsizeX@#1}
3729 \footsplitskips
3730 \setbox0=\vbox{\hsize=\maxdimen%
3731 \let\bidirTL@everypar\@empty%
3732 \noindent\color@begingroup%
3733 \csuse{bhooknoteX@#1}%
3734 \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3735 \setbox0=\hbox{\unvXH{0}{#1}}%
3736 \dp0=\z@
3737 \ht0=\csname footfudgefactor#1\endcsname\wd0
3738 \box0
3739 \penalty0}}
3740
3741 %

\unvXH42 \newcommand*{\unvXH}[2]{% 2th is optional for retro-compatibility
3743 \setbox0=\vbox{\unvbox#1%
3744 \global\setbox1=\lastbox}%
3745 \unhbox1
3746 \unskip % remove \rightskip,
3747 \unskip % remove \parfillskip,
3748 \unpenalty % remove \penalty of 10000,
3749 \hskip\csuse{afternoteX@#2}} % but add the glue to go between the notes

```

```
3750
3751 %
```

```
\parafootfmtX \parafootfmtX{<series>}
```

```
3752 \newcommand*{\parafootfmtX}[2]{%
3753   \protected@edef\@currentlabel{%
3754     \@nameuse{@thefnmark#1}%
3755   }%
3756   \insertparafootsepX{#1}%
3757   \ledsetnormalparstuff@common%
3758   {\csuse{notenumfontX@#1}%
3759    \csuse{notenumfontX@#1}%
3760    \wrapped@footfootmarkX{#1}%
3761    \strut%
3762    #2\penalty-10}}
3763
3764 %
```

```
\para@footgroupX \para@footgroupX{<series>}
\mppara@footgroupX
```

```
3765 \newcommand*{\para@footgroupX}[1]{%
3766   \hsize=\expandafter\dimexpr\csuse{hsizeX@#1}\relax%
3767   \unvbox\curname footins#1\endcurname
3768   \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3769   \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3770   \makehboxofhboxes
3771   \setbox0=\hbox{\unhbox0 \removehboxes}%
3772   \csuse{bhookgroupX@#1}
3773   \csuse{notefontsizeX@#1}
3774   \unhbox0\par}
3775
3776 \newcommand*{\mppara@footgroupX}[1]{%
3777   \setnoteswidthliketwocolumnsX@{#1}%
3778   \vskip\skip\@nameuse{mpfootins#1}
3779   \ifl@dpairing\ifparledgroup
3780     \leavevmode%
3781     \leavevmode\marks\parledgroup@{begin}%
3782     \marks\parledgroup@series{#1}%
3783     \marks\parledgroup@type{footnoteX}%
3784     \fi\fi\normalcolor
3785     \ifparledgroup%
3786       \ifl@dpairing%
3787         \else%
3788           \setnoteswidthliketwocolumnsX@{#1}%
3789           \setnotesXpositionliketwocolumns@{#1}%
3790           \print@footnoteXrule{#1}%
3791         \fi%
3792       \else%
3793         \setnoteswidthliketwocolumnsX@{#1}%
```

```

3794 \setnotesXpositionliketwocolumns@{#1}%
3795 \print@footnoteXrule{#1}%
3796 \fi%
3797 \unvbox\csname mpfootins#1\endcsname
3798 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3799 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3800 \makehboxofhboxes
3801 \setbox0=\hbox{\unhbox0 \removehboxes}%
3802 \csuse{bhookgroupX@#1}%
3803 \csuse{notefontsizeX@#1}%
3804 \unhbox0\par}}
3805
3806 %

```

**Insertion of the footnotes separator** The command `\insertparafootsepX{<series>}` must be called at the beginning of `\parafootftmX`.

```

\prevpage@num07 \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep08 \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3809 {\csuse{parafootsepX@#1}}%
3810 }%
3811 }
3812 %

```

### 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.

```

3813 \newcommand{\wrapped@footfootmarkX}[1]{%
3814 \ifdefined\hypertarget%
3815 \Hy@raisedlink{%
3816 \hypertarget%
3817 {\footnotemark#1@this@footnoteX@reading}%
3818 }%
3819 }%
3820 \hyperlink%
3821 {\@bodyfootmark#1@this@footnoteX@reading}%
3822 {\@nameuse{footfootmark#1}}%
3823 \else%
3824 \@nameuse{footfootmark#1}%
3825 \fi%
3826 }%
3827 %

```

`\wrapped@bodyfootmarkX` `\wrapped@bodyfootmarkX` prints the footnote mark of the text body, wrapped in `hyperref` package's commands, if needed.



```

3828 \newcommand{\wrapped@bodyfootmarkX}[1]{%
3829   \ifdefined\hypertarget%
3830     \Hy@raisedlink{%
3831       \hypertarget%
3832         {@bodyfootmark#1@\expandafter\the\csname footnote#1@reading\
3833         endcsname}%
3834       }%
3835     \hyperlink%
3836       {@footnotemark#1@\expandafter\the\csname footnote#1@reading\
3837       endcsname}%
3838     {\@nameuse{bodyfootmark#1}}%
3839   \else%
3840     \@nameuse{bodyfootmark#1}%
3841   \fi%
3842 }%

```

## 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<sup>La</sup>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 3843 \newcommand*{\ledsetnormalparstuff@common}{%
3844   \ifluatex%
3845     \textdir\footnote@luatextextdir%
3846     \pardir\footnote@luatexpardir%
3847     \fi%
3848     \csuse{\csuse{footnote@dir}}%
3849     \normal@pars%
3850     \parfillskip \z@ \@plus 1fil}%
3851
3852 \newcommand*{\Xledsetnormalparstuff}[1]{%
3853   \ledsetnormalparstuff@common%
3854   \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%
3855 }%
3856
3857 \newcommand*{\ledsetnormalparstuffX}[1]{%
3858   \ledsetnormalparstuff@common%
3859   \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%

```

```
3860 }%
3861 %
```

## 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. They set, if the option is enabled, the `\hsize`. They are also called at the on the setup for paragraphed notes.

`\setXnoteswidthliketwocolumns@`  
`\setnoteswidthliketwocolumnsX@`

```
3862
3863 \newdimen\old@hsize%
3864 \AtBeginDocument{\old@hsize=\hsize}%
3865
3866 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3867   \global\let\hsize@fornote=\hsize%
3868   \global\old@hsize=\hsize%
3869   \let\old@columnwidth=\columnwidth%
3870   \iftoggle{Xnoteswidthliketwocolumns@#1}%
3871     {%
3872       \csuse{setwidthliketwocolumns@\columns@position}%
3873       \global\let\hsize@fornote=\hsize%
3874     }%
3875   }%
3876   \let\hsize=\hsize@fornote%
3877   \let\columnwidth=\old@columnwidth%
3878 }%
3879
3880 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3881   \global\let\hsize@fornote=\hsize%
3882   \global\old@hsize=\hsize%
3883   \let\old@columnwidth=\columnwidth%
3884   \iftoggle{noteswidthliketwocolumnsX@#1}%
3885     {%
3886       \csuse{setwidthliketwocolumns@\columns@position}%
3887       \global\let\hsize@fornote=\hsize%
3888     }%
3889   }%
3890   \let\hsize=\hsize@fornote%
3891   \let\columnwidth=\old@columnwidth%
3892 }%
3893
3894 %
```

`\setXnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They

`\setnotesXpositionliketwocolumns@`

call commands which are defined only in `reledpar`, because this feature has no sens without `reledpar`.

```

3895 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3896   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3897     \csuse{setnotespositionliketwocolumns@\columns@position}%
3898   }{}%
3899 }%
3900
3901 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3902   \iftoggle{noteswidthliketwocolumnsX@#1}{%
3903     \csuse{setnotespositionliketwocolumns@\columns@position}%
3904   }{}%
3905 }%
3906
3907 %

```

## XVI Footnotes' order

`\fnpos` The `\fnpos` and `\mpfnpos` simply place their arguments in `\@fnpos` and `\@mpfnpos`, which will be used later in the output routine.

```

\fnpos
\mpfnpos
\@fnpos
\@mpfnpos
3908 \def\@fnpos{familiar-critical}
3909 \def\@mpfnpos{critical-familiar}
3910 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3911 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3912 %

```

## 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@Xfootnoterule13 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnoteXrule14 \vskip-\csuse{Xafterterrule@#1}%Because count in \dimen\csuse{#1footins}
3915 \nointerlineskip%
3916 \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
3917 \nointerlineskip%
3918 \vskip\csuse{Xafterterrule@#1}%
3919 }%
3920
3921 \newcommand{\print@footnoteXrule}[1]{%
3922   \vskip-\csuse{afterterruleX@#1}%Because count in \dimen\csuse{footins#1}
3923   \nointerlineskip%
3924   \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
3925   \nointerlineskip%
3926   \vskip\csuse{afterterruleX@#1}%

```

```

3927 }%
3928
3929 %

```

## 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 `\preXnotes` will be added before first series notes instead of the notes skip.

```

3930 \newtoggle{preXnotes@}
3931 \toggletrue{preXnotes@}
3932 \newcommand{\preXnotes@}{0pt}
3933 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3934 %

```

The same, but for familiar footnotes.

```

\preXnotes 35 \newtoggle{prenotesX@}
\preXnotes@ 36 \toggletrue{prenotesX@}
3937 \newcommand{\prenotesX@}{Opt}
3938 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
3939 %

```

### XVIII.0.3 Internal commands

```

firstXseries@ 40 \gdef\firstXseries@{}
prepare@preXnotes 41 \newcommand{\prepare@preXnotes}[1]{%
3942 \ifdimequal{Opt}{\preXnotes@}%
3943 {}%
3944 {%
3945 \IfStrEq{\firstXseries@}{}{%
3946 \global\skip\csuse{#1footins}=\preXnotes@%
3947 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3948 \gdef\firstXseries@{#1}%
3949 }%
3950 {%
3951 \ifseriesbefore{#1}{\firstXseries@}%
3952 {%
3953 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3954 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3955 \gdef\firstXseries@{#1}%
3956 }%
3957 {%
3958 }%
3959 }%
3960 }
3961 %

```

The same thing is required for familiar notes and \prenotesX.

```

firstseriesX@ 62 \gdef\firstseriesX@{}
prepare@prenotesX 63 \newcommand{\prepare@prenotesX}[1]{%
3964 \ifdimequal{Opt}{\prenotesX@}%
3965 {}%
3966 {%
3967 \IfStrEq{\firstseriesX@}{}{%
3968 \global\skip\csuse{footins#1}=\prenotesX@%
3969 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3970 \gdef\firstseriesX@{#1}%
3971 }%

```

```

3972     {%
3973     \ifseriesbefore{#1}{\firstseriesX@}%
3974     {%
3975     \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
3976     \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3977     \gdef\firstXseries@{#1}%
3978     }%
3979     {%
3980     }%
3981     }%
3982 }
3983 %

```

## XIX Endnotes

First, check the noend option.

```

3984 \ifbool{noend@}{}{%Used instead of \ifnoend@ to prevent expansion problem
3985 %

```

**\l@dend@open** **\l@dend@close** **\l@dend@stuff** **\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.

```

3986 \newcommand{\l@dend@open}[1]{%
3987 \global\booltrue{l@dend@#1}%
3988 \expandafter\immediate%
3989 \expandafter\openout%
3990 \csname l@d@#1end\endcsname%
3991 =\jobname.#1end\relax%
3992 }%
3993 \newcommand{\l@dend@close}[1]{%
3994 \global\boolfalse{l@dend@#1}%
3995 \expandafter\immediate%
3996 \expandafter\closeout\csname l@d@#1end\endcsname%
3997 }%
3998
3999 %

```

**\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.

```

4000 \newcommand{\l@dend@stuff}{%
4001 \def\do##1{%

```

```

4002 \ifbool{l@dend@##1}{}%
4003   {\l@dend@open{##1}}%
4004   \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{\the\section@num}}%
4005   }%
4006   \dolistloop{@series}%
4007 }%
4008
4009 %

```

`\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.

```

4010 \global\notbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5#6#7{%
4011   \hangindent=\csuse{Xendhangindent@#4}%
4012   \ifXendinsertsep@%
4013     \hskip\csuse{Xendafternote@#4}%
4014     \csuse{Xendsep@#4}%
4015   \else%
4016     \iftoggle{Xendparagraph@#4}%
4017       {\global\Xendinsertsep@true}%
4018     {}%
4019   \fi%
4020   \xdef\@currentseries{#4}%
4021   \def\do##1{%
4022     \setkeys[mac]{truefootnoteoption}{##1}%
4023   }%
4024   \notblank{#5}{\docsvlist{#5}}{%
4025     \csuse{Xendbhooknote@#4}%
4026     \csuse{Xendnotefontsize@#4}%
4027     \IfStrEq{#6}{R}{\ledRcol@true}{}%
4028     \def\@this@crossref@start{#7:start}%
4029     \def\@this@crossref@end{#7:end}%

```

```

4030 \printlineendnote{#1}{#4}%
4031 \IfStrEq{#6}{R}{\ledRcol@false}{}%
4032 \undef\@this@crossref@start%
4033 \undef\@this@crossref@end%
4034 \nottoggle{Xendlemmadisablefontselection@#4}%
4035 {\select@lemmafont#1|{\csuse{Xendlemmafont@#4}#2}}%
4036 {\{\csuse{Xendlemmafont@#4}#2}}%
4037 \ifboolexpr{%
4038   togl {nosep@}%
4039   or test{\ifcseempty{Xendlemmaseparator@#4}}%
4040 }%
4041 {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
4042 {\nobreak%
4043   \hskip\csuse{Xendbeforelemmaseparator@#4}%
4044   \csuse{Xendlemmaseparator@#4}%
4045   \hskip\csuse{Xendafterlemmaseparator@#4}%
4046 }%
4047 #3%
4048 \nottoggle{Xendparagraph@#4}{\par}{}%
4049 \def\do##1{%
4050   \setkeys[mac]{falsefootnoteoption}{##1}%
4051 }%
4052 \notblank{#5}{\docsvlist{#5}}{}%
4053 }}%
4054
4055 \let\l@d@section=\@gobble
4056
4057 %

```

**\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.

```

4058 \newcommand{\printlineendnote}[2]{%
4059   \l@dp@rsefootspec#1|%
4060   \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
4061     \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4062       \l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
4063       \l@dparsedendsub}%
4064     }%
4065     {\
4066       \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
4067       \l@dparsedstartsub}%
4068     }%
4069     \ifboolexpr{%
4070       togl {nonum@}%
4071       or togl {Xendnonumber@#2}}%
4072     {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
4073     {

```



```

4072 \iftoggle{Xendnumberonlyfirstinline@#2}%
4073 {\ifcsdef{prevendline#2}%
4074 {\ifcsequal{prevendline#2}{lineinfo@}%
4075 {%
4076 \csuse{Xendbhookinplaceofnumber@#2}%
4077 \ifcsequal{Xendsymmlinenumber@#2}%
4078 {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
4079 {\printsymmlineendnotearea{#2}}%
4080 \csuse{Xendahookinplaceofnumber@#2}%
4081 }%
4082 {\printlineendnotearea{#1}{#2}}}%
4083 {\printlineendnotearea{#1}{#2}}}%
4084 }%
4085 {\printlineendnotearea{#1}{#2}}%We keep every time line
4086 \csxdef{prevendline#2}{\lineinfo@}%
4087 }%
4088 }%
4089 %

```

```

\printsymmlineendnotearea \newcommand{\printsymmlineendnotearea}[1]{%
4091 \hspace{\csuse{Xendbeforemlinenumber@#1}}%
4092 \csuse{Xendnotenumfont@#1}%
4093 \ifdimequal{\csuse{Xendboxmlinenumber@#1}}{\z@}%
4094 {\csuse{Xendsymmlinenumber@#1}}%
4095 {\hbox to \csuse{Xendboxmlinenumber@#1}%
4096 {\csuse{Xendsymmlinenumber@#1}\hfill}%
4097 }%
4098 \hspace{\csuse{Xendaftersymmlinenumber@#1}}%
4099 }%
4100 %

```

**\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.)

```

4101 \newcommand{\printlineendnotearea}[2]{%
4102 \csuse{Xendbhooklinenumber@#2}%
4103 \hspace{\csuse{Xendbeforenumber@#2}}%
4104 \bgroup%
4105 \csuse{Xendnotenumfont@#2}%
4106 \ifdimequal{\csuse{Xendboxlinenumber@#2}}{\Opt}%
4107 {\printendlines#1||\ifledRcol@{\@Rlineflag\fi}%
4108 {\leavevmode%
4109 \hbox to \csuse{Xendboxlinenumber@#2}%
4110 {%
4111 \IfSubStr{RC}{\csuse{Xendboxlinenumberalign@#2}}{\hfill}{}}%
4112 \printendlines#1||\ifledRcol@{\@Rlineflag\fi}%
4113 \IfSubStr{LC}{\csuse{Xendboxlinenumberalign@#2}}{\hfill}{}}%

```

```

4114 }}%
4115 \egroup%
4116 \hspace{\csuse{Xendafternumber@#2}}%
4117 \csuse{Xendahooklinenumber@#2}%
4118 }%
4119 %

```

**\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.

```

4120 \newif\ifXendinsertsep@%
4121 \newcommand*\doendnotes}[1]{%
4122   \l@dend@close{#1}%
4123   \beginngroup
4124     \makeatletter
4125     \expandafter\let\csname #1end\endcsname=\endprint
4126     \input\jobname.#1end%
4127     \global\Xendinsertsep@false%
4128   \endgroup}
4129 %

```

**\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.

```

4130 \newcommand*\doendnotesbysection}[1]{%
4131   \l@dend@close{#1}%
4132   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
4133   \beginngroup%
4134     \makeatletter%
4135     \def\l@d@section##1{%
4136       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
4137       {\cslet{#1end}{\endprint}}%
4138       {\cslet{#1end}{\@gobblefive}}%
4139     }%
4140     \input\jobname.#1end%
4141     \global\Xendinsertsep@false%
4142   \endgroup%
4143 }%
4144 %

```

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.

```

4145 }%
4146 %

```

`\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.

```
4147 \newcommand*{\setprintendlines}[6]{%
4148   \l@dpnumfalse \l@ddashfalse
4149   \ifnum#4=#1 \else
4150     \l@dpnumtrue
4151     \l@ddashtrue
4152   \fi
4153   %
```

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.

```
4154   \ifl@dpnum \l@delintrue \else \l@delinfalse \fi
4155   \ifnum#2=#5 \else
4156     \l@delintrue
4157     \l@ddashtrue
4158   \fi
4159   %
```

We print the starting sub-line if it is nonzero.

```
4160   \l@dsesubfalse
4161   \ifnum#3=0 \else
4162     \l@dsesubtrue
4163   \fi
4164   %
```

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.

```
4165   \l@deselfalse
4166   \ifnum#6=0 \else
4167     \ifnum#6=#3
4168       \ifl@delin \l@deseltrue \else \l@deselfalse \fi
4169     \else
4170       \l@deseltrue
4171       \l@ddashtrue
4172     \fi
4173   \fi%
4174   %
4175   \ifl@ddash%
```

```

4176 \ifbool{expr{togl{fulllines@} or test{\ifcempty{Xendtwolines@}\
@currentseries}}}%
4177 {}%
4178 {%
4179 \setistwofollowinglines{#1}{#2}{#4}{#5}%
4180 \ifbool{expr{
4181 (%
4182 togl {Xendtwolinesbutnotmore@}\@currentseries}%
4183 and not%
4184 (%
4185 bool {istwofollowinglines@}%
4186 )}%
4187 )%
4188 or%
4189 (%
4190 (not test{\ifnumequal{#1}{#4}})%
4191 and togl{Xendtwolinesonlyinsamepage@}\@currentseries}%
4192 )%
4193 }%
4194 {}%
4195 {%
4196 \l@d@dashfalse%
4197 \l@d@Xtwolinestrue%
4198 \l@d@elinfalse%
4199 \l@d@eslfalse%
4200 \ifcempty{Xendmorethantwolines@}\@currentseries}%
4201 {}%
4202 {\ifistwofollowinglines@}\else%
4203 \l@d@Xmorethantwolinestrue%
4204 \fi%
4205 }%
4206 }%
4207 }%
4208 \fi%
4209 %

```

End of \setprintendlines.

```

4210 }%
4211 %

```

**\printendlines** Now we are ready to print it all.

```

4212 \def\printendlines#1|#2|#3|#4|#5|#6|#7|#8|{\begingroup
4213 \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
4214 %

```

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.

```

4215 \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
4216   {\bgroup}%
4217   {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%
4218   \wrap@edcrossref{\@this@crossref@start}{\printnpnum{#1}}%
4219   \ifl@d@dash%
4220   \ifl@d@pnum%
4221     \csuse{Xendlineprefixsingle@\@currentseries}%
4222   \else%
4223     \ifcsempy{Xendlineprefixmore@\@currentseries}%
4224       {\csuse{Xendlineprefixsingle@\@currentseries}}
4225       {\csuse{Xendlineprefixmore@\@currentseries}}%
4226   \fi%
4227 \else%
4228   \csuse{Xendlineprefixsingle@\@currentseries}%
4229 \fi%
4230 \wrap@edcrossref{\@this@crossref@start}{\linenumrep{#2}}%
4231 \iftoggle{Xendlineflag@\@currentseries}{\ifledRcol@\@Rlineflag\fi}{}%
4232 \ifl@d@ssub%
4233   \csuse{Xendsublinesep@\@currentseries}%
4234   \wrap@edcrossref{\@this@crossref@start}{\sublinenumrep{#3}}%
4235 \fi%
4236 \egroup%
4237 %

```

And now, print the dash + the end line number, or the line number range symbol.

```

4238 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
4239   {\bgroup}%
4240   {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
4241   \ifl@d@Xtwolines%
4242     \ifl@d@Xmorethantwolines%
4243       \csuse{Xendmorethantwolines@\@currentseries}%
4244     \else%
4245       \csuse{Xendtwolines@\@currentseries}%
4246     \fi%
4247   \else%
4248     \ifl@d@dash%
4249       \ifdefined\linerangesep%
4250         \linerangesep%
4251       \else%
4252         \csuse{Xendlinerangeseparator@\@currentseries}%
4253       \fi%
4254     \fi%
4255     \ifl@d@pnum%
4256       \wrap@edcrossref{\@this@crossref@end}{\printnpnum{#4}}%
4257     \fi%
4258     \ifl@d@elin%
4259       \ifl@d@pnum\csuse{Xendlineprefixsingle@\@currentseries}\fi%

```

```

4260 \wrap@edcrossref{\@this@crossref@end}{\linenumrep{#5}}%
4261 \iftoggle{Xendlineflag@{\@currentseries}{\ifledRcol@{\@Rlineflag\fi}{}}%
4262 \fi%
4263 \ifl@d@esl%
4264 \ifl@d@elin%
4265 \csuse{Xendsublinesep@{\@currentseries}}%
4266 \fi%
4267 \wrap@edcrossref{\@this@crossref@end}{\sublinenumrep{#6}}%
4268 \fi%
4269 \fi%
4270 \ifdimequal{\csuse{Xendboxendlinenum@{\@currentseries}}{0pt}}%
4271 {}%
4272 {\hfill}%Prevent underfull hbox
4273 \egroup%
4274 \endgroup%
4275 }%
4276
4277 %

```

**\printnpnum** A macro to print a page number in an endnote. Should not be override anymore

```

4278 \newcommand*{\printnpnum}[1]{\csuse{Xendbeforepagenumber@{\@currentseries}
4279 }#1\csuse{Xendafterpagenumber@{\@currentseries}}
4280 %

```

## XX **Generate series of notes**

In this section, X means the name of the series (A, B etc.)

**\series** \series\series creates one more new series. It is a public command, which just loops on the private command \newseries@.

```

4281 \newcommand{\newseries}[1]{%
4282 \def\do##1{\newseries@{##1}}%
4283 \docsvlist{#1}
4284 }
4285 %

```

**\@series** The \series@ macro is an etoolbox list, which contains the name of all series.

```

4286 \newcommand{\@series}{}
4287 %

```

The command \newseries@\series creates a new series of the footnote.

**\newseries@** \newcommand{\newseries@}[1]{  
%

## XX.1 Test if series is still existing

```

4290 \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
4291 {%
4292 %

```

## XX.2 Init specific to reledpar

When calling `\newseries@` after having loaded `reledpar`, we need to load specific setting.

```

4293 \ifdefined\newseries@par%
4294 \newseries@par{#1}%
4295 \fi%
4296 %

```

## XX.3 For critical footnotes

Critical footnotes are those which start with letters. We look for the `\nocritical` option of `reledmac`.

```

4297 \unless\ifnocritical@
4298 %

```

### XX.3.1 Options

```

4299 \newtoggle{Xlineflag@#1}
4300 \newtoggle{Xparindent@#1}
4301 \newtoggle{Xlemmadisablefontselection@#1}
4302 \csgdef{Xhangindent@#1}{Opt}%
4303 \csgdef{Xragged@#1}{}%
4304 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
4305 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
4306 \csgdef{Xcolalign@#1}{\raggedright}%
4307 \csgdef{Xnotenumfont@#1}{\normalfont}%
4308 \csgdef{Xnotefontsize@#1}{\footnotesize}%
4309 \csgdef{Xbhooknote@#1}{}%
4310 \csgdef{Xbhookgroup@#1}{}%
4311
4312 \csgdef{Xboxlinenum@#1}{Opt}%
4313 \csgdef{Xboxlinenumalign@#1}{L}%
4314
4315 \csgdef{Xboxstartlinenum@#1}{Opt}%
4316 \csgdef{Xboxendlinenum@#1}{Opt}%
4317
4318 \csgdef{Xboxsymlinenum@#1}{Opt}%
4319 \newtoggle{Xnumberonlyfirstinline@#1}%
4320 \newtoggle{Xnumberonlyfirstintwolines@#1}%
4321 \csgdef{Xtwolines@#1}{}%

```

```

4322 \csgdef{Xmorethantwolines@#1}{}%
4323 \csgdef{Xsublinesep@#1}{\fullstop}%
4324 \newtoggle{Xtwolinesbutnotmore@#1}%
4325 \newtoggle{Xtwolinesonlyinsamepage@#1}%
4326 \newtoggle{Xonlypstart@#1}%
4327 \newtoggle{Xpstarteverytime@#1}%
4328 \newtoggle{Xpstart@#1}%
4329 \newtoggle{Xstanza@#1}%
4330 \csgdef{Xstanzaseparator@#1}{}%
4331 \csgdef{Xsymlinenum@#1}{}%
4332 \newtoggle{Xnonumber@#1}%
4333 \csgdef{Xbeforenumber@#1}{0pt}%
4334 \csgdef{Xtxtbeforenumber@#1}{}%
4335 \csgdef{Xafternumber@#1}{0.5em}%
4336 \newtoggle{Xnonbreakableafternumber@#1}%
4337 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
4338 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
4339 \csgdef{Xinplaceofnumber@#1}{1em}%
4340 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
4341 \csgdef{Xbeforelemmaseparator@#1}{0em}%
4342 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
4343 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
4344 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}%
4345 \csgdef{Xafterrule@#1}{0pt}%
4346 \csgdef{Xtxtbeforenotes@#1}{}%
4347 \csgdef{Xmaxhnotes@#1}{0.8\vsizel}%
4348 \newtoggle{Xnoteswidthliketwocolumns@#1}%
4349 \csgdef{Xparafootsep@#1}{}%
4350 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}%
4351 \csgdef{Xlinerrangeseparator@#1}{\endashchar}%
4352
4353 \csgdef{Xlemmafont@#1}{}%
4354 \csgdef{Xhsize@#1}{\hsize}%
4355 %

```

### XX.3.2 Create inserts, needed to add notes in foot

As regards inserts, see chapter 15 of *The TeXbook* by D. Knuth.

```

4356 \expandafter\newinsert\csname #1footins\endcsname%
4357 \unless\ifnoledgroup%
4358 \expandafter\newinsert\csname mp#1footins\endcsname%
4359 \fi%
4360 %

```

### 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.



```

4361 \global\newcommand\parapparus@{\expandafter\newcommand\expandafter
*}{\expandafter\newcommand\csname #1footnote\endcsname[2] [] {%
4362 \ifnum\@edtext@level>0%
4363 \begingroup%
4364 \newcommand{\content}{##2}%
4365 \ifnumberedpar@%
4366 \ifledRcol%
4367 \ifluatex%
4368 \footnotelang@lua[R]%
4369 \fi%
4370 \@ifundefined{xpg@main@language}%if polyglossia
4371 {}%
4372 {\footnotelang@poly[R]}%
4373 \footnoteoptions@{R}{##1}{true}%
4374 \xright@appenditem{%
4375 \ifbool{indtl@innote}%
4376 {\unexpanded{\let\index\nindex}}%
4377 {}%
4378 \ifbool{indtl@notenumber}%
4379 {\unexpanded{\let\index\nindex}}%There is no note
...number so
4380 {}%
4381 \noexpand\Xnote@true%
4382 \noexpand\prepare@preXnotes{##1}%
4383 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4384 \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.
4385 \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4386 \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
4387 \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
4388 \noexpand\csuse{v#1footnote}{##1}%
4389 {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4390 \noexpand\Xnote@false%
4391 \unexpanded{\undef\@this@crossref@start}%
4392 \unexpanded{\undef\@this@crossref@end}%
4393 \ifbool{indtl@innote}%
4394 {\unexpanded{\let\index\orig@@index}}%
4395 {}%
4396 \ifbool{indtl@notenumber}%
4397 {\unexpanded{\let\index\orig@@index}}%
4398 {}%
4399 }\to\inserts@listR
4400 \footnoteoptions@{R}{##1}{false}%
4401 \global\advance\insert@countR \@ne%
4402 \else%

```

```

4403         \ifluatex%
4404             \footnotelang@lua%
4405         \fi%
4406         \@ifundefined{xpg@main@language}%if polyglossia
4407             {}%
4408             {\footnotelang@poly}%
4409         \footnoteoptions@{L}{##1}{true}%
4410         \xright@appenditem{%
4411             \ifbool{indtl@innote}%
4412                 {\unexpanded{\let\index\nindex}}%
4413                 {}%
4414             \ifbool{indtl@notenumber}%
4415                 {\unexpanded{\let\index\nindex}}%There is no note
...number so
4416             {}%
4417             \noexpand\Xnote@true%
4418             \noexpand\prepare@preXnotes{#1}%
4419             \noexpand\prepare@edindex@fornote{\l@d@nums}%
4420             \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.
4421             \ifl@dpairing%
4422                 \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4423             \fi%
4424             \unexpanded{\def\@this@crossref@start}{\theedtext:
start}%
4425             \unexpanded{\def\@this@crossref@end}{\theedtext:end}%
4426             \noexpand\csuse{v#1footnote}%
4427                 {#1}%
4428                 {\l@d@nums}{\expandonce\@tag}{\expandonce\content
}}%
4429             \unexpanded{\undef\@this@crossref@start}%
4430             \unexpanded{\undef\@this@crossref@end}%
4431             \noexpand\Xnote@false%
4432             \ifbool{indtl@innote}%
4433                 {\unexpanded{\let\index\orig@@index}}%
4434                 {}%
4435             \ifbool{indtl@notenumber}%
4436                 {\unexpanded{\let\index\orig@@index}}%
4437                 {}%
4438             }\to\inserts@list
4439             \global\advance\insert@count \@ne%
4440             \footnoteoptions@{L}{##1}{false}%
4441         \fi
4442     \else
4443         \csuse{v#1footnote}{#1}{\{0|0|0|0|0|0|0\}}{##1}%
4444     \fi%
4445 \endgroup%
```

```

4446         \else%
4447             \led@err@FootnoteWithoutEdtext%
4448         \fi%
4449     \ignorespaces%
4450     }
4451 %

```

We need to be able to modify reledmac's footnote macros and restore their

```

4452     \global\csletcs{#1@@footnote}{#1footnote}
4453 %

```

### XX.3.4 Set standard display

```

4454     \Xarrangement@normal{#1}%
4455 %

```

End of for critical footnotes.

```

4456     \fi
4457 %

```

## XX.4 For familiar footnotes

Familiar footnotes are those which end with letters. We look for the `nofamiliar` option of `reledmac`.

```

4458     \unless\ifnofamiliar@
4459 %

```

### XX.4.1 Options

```

4460     \newtoggle{parindentX@#1}
4461     \csgdef{hangindentX@#1}{Opt}%
4462     \csgdef{raggedX@#1}{}%
4463     \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
4464     \csgdef{hsizethreecolX@#1}{.3 \hsize}%
4465     \csgdef{colalignX@#1}{\raggedright}%
4466     \csgdef{notenumfontX@#1}{\normalfont}%
4467     \csgdef{notefontsizeX@#1}{\footnotesize}%
4468     \csgdef{bhooknoteX@#1}{}%
4469     \csgdef{bhookgroupX@#1}{}%
4470     \csgdef{afterruleX@#1}{Opt}
4471     \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
4472     \csgdef{maxhnotesX@#1}{0.8\vsizex}%
4473     \newtoggle{noteswidthliketwocolumnsX@#1}%
4474     \csgdef{parafootsepX@#1}{}%
4475     \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4476     \csgdef{hsizexX@#1}{\hsizex}%
4477 % End of for familiar footnotes.

```

```

4478 % \subsubsection{Create inserts, needed to add notes in foot}
4479 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4480 % \begin{macrocode}
4481 \expandafter\newinsert\csname footins#1\endcsname%
4482 \unless\ifnoledgroup%
4483 \expandafter\newinsert\csname mpfootins#1\endcsname%
4484 \fi%
4485 %

```

#### 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.

```

4486 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4487 \begingroup%
4488 \prepare@prenotesX{#1}%
4489 \newcommand{\content}{##1}%
4490 %
4491 %

```

If we are preparing parallel typesetting, we cannot just increase the footnote counter. Read `reledpar`'s handbook about that (V.2.1 p. 44).

```

4492 \global\expandafter\advance\csname footnote#1@reading\
endcsname by \@ne%
4493 \ifl@dpairing%
4494 \ifcsdef{footnote#1reading\the\csname footnote#1@reading\
endcsname=typeset}%
4495 {\setcounter{footnote#1}{\csuse{footnote#1reading\the\
csname footnote#1@reading\endcsname=typeset}}}%
4496 {\setcounter{footnote#1}{\the\csname footnote#1@reading
\endcsname}}}%
4497 \else%
4498 \stepcounter{footnote#1}%
4499 \fi%
4500 %

```

And now, the feature not depending of whether we are preparing parallel typesetting

```

4501 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4502 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4503 {\csuse{@footnotemark#1}}%
4504 {}%
4505 \ifluatex%
4506 \xdef\footnote@luatextextdir{\the\textdir}%
4507 \xdef\footnote@luatexpardir{\the\pardir}%
4508 \fi%
4509 \if@ledgroup%
4510 \led@set@index@fornote{#1}%
4511 \fi%
4512 \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%

```

```

4513         \ifbool{indtl@innote}%
4514             {\let\index\orig@index}%
4515             {}%
4516         \ifbool{indtl@notenumber}%
4517             {\let\index\orig@index}%
4518             {}%
4519         \endgroup%
4520     }
4521 %

```

Then define the counters. The  $\text{\TeX}$  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. 44).

```

4522     \newcounter{footnote#1}
4523     \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4524     \expandafter\newcount\csname footnote#1@reading\endcsname%
4525 %

```

Do not forget to initialize series

```

4526     \arrangementX@normal{#1}%
4527     \fi
4528 %

```

## XX.5 The endnotes

Endnotes are commands like `\Xendnote`, where `X` is a series letter. First, we check for the `noend` options.

```

4529     \unless\ifnoend@
4530 %

```

### XX.5.1 The auxiliary file

Endnotes of all varieties are saved up in a file, one by series, typically named `\jobname.Xend`. `\l@d@Xend` is the output stream number for this file, and `\ifl@dend@X` is a flag that is true when the file is open.

```

\l@d@Xend
\ifl@dend@X
\l@dend@Xtrue
\l@dend@Xfalse
4531     \expandafter\newwrite\csname l@d@#1end\endcsname%
4532     \expandafter\newif\csname ifl@dend@#1\endcsname%
4533 %

```

### 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.

```

4534 \global\expandafter\newcommand\csname #1endnote\endcsname[2][1,
4535 usedefault]{%
4536 \bgroup%
4537 \newlinechar='40%
4538 \global\@noneed@Footnotetrue%
4539 \newcommand{\content}{##2}%
4540 \stepcounter{labidx}%
4541 \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
4542 \expandafter\string\csname #1end\endcsname%
4543 {\ifnumberedpar@l@d@nums\fi}%
4544 {\ifnumberedpar@\expandonce\@tag\fi}%
4545 {\expandonce\content}%
4546 {\#1}%
4547 {\unexpanded{##1}}%
4548 {\ifledRcol R\else L\fi}%
4549 {\theedtext}%
4550 \@percentchar%
4551 }%
4552 \egroup%
4553 \ignorespaces%
4554 }%
4555 %

```

\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.

```

4556 \global\cslet{#1end}{\@gobblefive}
4557 %
4558 %

```

We need to store the number of times \doendnotesbysection is called for one series.

```

4559 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4560 %

```

### XX.5.3 The options

```

4561 \csgdef{Xendtwolines@#1}{}%
4562 \csgdef{Xendmoreethantwolines@#1}{}%
4563 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4564 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4565 \newtoggle{Xendlemmadisablefontselection@#1}{}%
4566 \csgdef{Xendnotenunfont@#1}{\normalfont}%
4567 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4568 \csgdef{Xendbhooknote@#1}{}%

```

```

4569 \csgdef{Xendsublinesep@#1}{\fullstop}%
4570
4571 \csgdef{Xendbeforenumber@#1}{Opt}
4572 \csgdef{Xendafternumber@#1}{0.5em}
4573
4574 \csgdef{Xendboxlinenum@#1}{Opt}%
4575 \csgdef{Xendboxlinenumalign@#1}{L}%
4576
4577 \csgdef{Xendboxstartlinenum@#1}{Opt}%
4578 \csgdef{Xendboxendlinenum@#1}{Opt}%
4579
4580 \csgdef{Xendlemmaseparator@#1}{}%
4581 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4582 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4583 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4584
4585 \newtoggle{Xendparagraph@#1}%
4586 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4587 \csgdef{Xendsep@#1}{}%
4588
4589 \csgdef{Xendinplaceofnumber@#1}{Opt}%
4590 \newtoggle{Xendnonumber@#1}%
4591
4592 \csgdef{Xendhangindent@#1}{Opt}%
4593 \newtoggle{Xendnumberonlyfirstinline@#1}%
4594 \newtoggle{Xendnumberonlyfirstintwolines@#1}%
4595
4596 \csgdef{Xendbeforesymlinenum@#1}{\csuse{Xendbeforenumber@#1}}%
4597 \csgdef{Xendaftersymlinenum@#1}{\csuse{Xendafternumber@#1}}%
4598 \csgdef{Xendsymlinenum@#1}{}%
4599 \csgdef{Xendboxsymlinenum@#1}{Opt}%
4600
4601 \csgdef{Xendbhooklinenum@#1}{}%
4602 \csgdef{Xendehooklinenum@#1}{}%
4603 \csgdef{Xendbhookinplaceofnumber@#1}{}%
4604 \csgdef{Xendehookinplaceofnumber@#1}{}%
4605
4606 \csgdef{Xendlinangeseparator@#1}{\endashchar}%
4607
4608 \csgdef{Xendbeforepagenumber@#1}{p.}%
4609 \csgdef{Xendafterpagenumber@#1}{) }%
4610 \csgdef{Xendlineprefixsingle@#1}{}%
4611 \csgdef{Xendlineprefixmore@#1}{}%
4612
4613 \newtoggle{Xendlineflag@#1}
4614
4615 \csgdef{Xendlemmafont@#1}{}%
4616 %
4617 %

```

End of endnotes declaration

```
4618 \fi%
4619 %
```

Dump series in \@series

```
4620 \listxadd{\@series}{#1}
4621 }
4622 }% End of \newseries
4623 %
```

## XX.6 Init standards series (A,B,C,D,E)

```
4624 \expandafter\newseries\expandafter{\default@series}
4625 %
```

## 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.

```
4626 \newcommand{\seriesatbegin}[1]{%
4627   \StrDel{\@series}{#1}[\@series]%
4628   \edef\@new{%
4629     \listead{\@new}{#1}%
4630     \listead{\@new}{\@series}%
4631     \xdef\@series{\@new}%
4632   }
4633 %
```

**\seriesatend** And `\seriesatend` moves the series to the end of the list.

```
4634 \newcommand{\seriesatend}[1]{%
4635   \StrDel{\@series}{#1}[\@series]%
4636   \edef\@new{%
4637     \listead{\@new}{\@series}%
4638     \listead{\@new}{#1}%
4639     \xdef\@series{\@new}%
4640   }
4641 %
```

### XXI.2 Test series order

**\ifseriesbefore** `\ifseriesbefore{<seriesA>}{<seriesB>}{<true>}{<false>}` expands `<true>` if `<seriesA>` is printed before `<seriesB>`, expands `<false>` otherwise.



```

4642 \newcommand{\ifseriesbefore}[4]{%
4643   \StrPosition{\@series}{#1}[\@first]%
4644   \StrPosition{\@series}{#2}[\@second]%
4645   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4646 }
4647 %

```

### XXI.2.1 Get the first series

In some specific case, we need to know the first series of the list of series.

```

\@getfirstseries@S 48 \newcommand{\@getfirstseries}{%
4649   \ifdefempty{\@series}%
4650   {\xdef\@firstseries{}}%
4651   {\StrChar{\@series}{1}[\@firstseries]}%
4652 }%
4653 %

```

## 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`.

```

4654 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4655   \def\do##1{%
4656     \global\settoggle{#2@##1}{#3}%
4657     \ifstrequal{#4}{critical}{
4658       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
4659     }{}
4660     \ifstrequal{#4}{familiar}{
4661       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
4662     }{}
4663   }%
4664   \ifstreempty{#1}{%
4665     \dolistloop{\@series}%
4666     \ifstreempty{#5}{}%
4667       \docsvlist{#5}%
4668     }
4669   }%
4670   {%
4671     \docsvlist{#1}%
4672   }%
4673 }
4674 %

```

`\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`.

```

4675 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4676   \def\do##1{
4677     \csgdef{#2@##1}{#3}
4678     \ifstrequal{#4}{critical}{%
4679       \csuse{Xarrangement@}\csuse{series@display##1}}{##1}%
4680     }{}
4681     \ifstrequal{#4}{familiar}{%
4682       \csuse{arrangementX@}\csuse{series@displayX##1}}{##1}%
4683     }{}%
4684   }%
4685   \ifstreempty{#1}{%
4686     \dolistloop{\@series}%
4687     \ifstreempty{#5}{}%
4688     \docsvlist{#5}
4689   }
4690   }%
4691   {%
4692     \docsvlist{#1}%
4693   }%
4694 }%
4695 %

```

### XXI.3.3 Tools to generate options commands

**\newhookcommand@series** `\newhookcommand@series\command names` is a generic command to add new commands for hooks, like `\Xhsizetwocol`. The first argument is the name of the hook, the second a comma-separated list of pseudo-series where the hook can be used, like `appref` in the case of `\Xtwolines`. The second argument is also used to create commands named `\<hookname><pseudoseris>`, like `\Xtwolinesappref`.

```

4696 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
4697   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4698     \setcommand@series{##1}{#1}{##2}[] [#2]%
4699   }%
4700   \ifstreempty{#2}{}%
4701   \def\do##1{%
4702     \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
4703     [1]{%
4704       \csuse{#1}[##1]{####1}%
4705     }%
4706     \docsvlist{#2}%
4707   }%
4708 }%
4709 %

```

**\newhooktoggle@series** `\newhooktoggle@series\command names` is a generic command to add new commands for a new toggle hook, like `\Xnumberonlyfirstinline`. The second argu-

ment is also used to create commands named `\<hookname><pseudoseris>`, like `\Xtwolinesbutnotmoreappref`.

```

4710 \newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
4711   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
4712     \settoggle@series{##1}{#1}{##2}[][#2]%
4713   }%
4714   \ifstrempy{#2}{%
4715     \def\do##1{%
4716       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
4717         \csuse{#1}[##1]%
4718       }%
4719     }%
4720     \docsvlist{#2}%
4721   }%
4722 }
4723 %

```

`\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 of notes

```

4724 \newcommand{\newhooktoggle@series@reload}[2]{%
4725   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
true},usedefault]{%
4726     \settoggle@series{##1}{#1}{##2}[#2]%
4727   }%
4728 }%
4729 %

```

`\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.

```

4730 \newcommand{\newhookcommand@series@reload}[2]{%
4731   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4732     \setcommand@series{##1}{#1}{##2}[#2]%
4733   }%
4734 }
4735 %

```

### 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.

```

4736 \unless\ifnocritical@
4737   \newhookcommand@series{Xlemmafont}%
4738   \newhooktoggle@series{Xparindent}

```

```

4739 \newhookcommand@series{Xhangindent}
4740 \newhookcommand@series{Xragged}
4741 \newhookcommand@series{Xhsizetwocol}
4742 \newhookcommand@series{Xhsizethreecol}
4743 \newhookcommand@series{Xcolalign}%
4744 \newhookcommand@series{Xnotenumfont}
4745 \newhookcommand@series{Xbhooknote}
4746 \newhookcommand@series@reload{Xbhookgroup}{critical}
4747 \newhookcommand@series{Xboxsymlinenum}%
4748 \newhookcommand@series{Xsymlinenum}
4749 \newhookcommand@series{Xbeforenumber}
4750 \newhookcommand@series{Xtxtbeforenumber}
4751 \newhookcommand@series{Xafternumber}
4752 \newhookcommand@series{Xbeforesymlinenum}
4753 \newhookcommand@series{Xaftersymlinenum}
4754 \newhookcommand@series{Xinplaceofnumber}
4755 \newhookcommand@series{Xlemmaseparator}
4756 \newhookcommand@series{Xbeforelemmaseparator}
4757 \newhookcommand@series{Xafterlemmaseparator}
4758 \newhookcommand@series{Xinplaceoflemmaseparator}
4759 \newhookcommand@series{Xtxtbeforenotes}
4760 \newhookcommand@series@reload{Xafterrule}{critical}
4761 \newhooktoggle@series{Xnumberonlyfirstinline}
4762 \newhooktoggle@series{Xnumberonlyfirstintwolines}
4763 \newhooktoggle@series{Xnonumber}
4764 \newhooktoggle@series{Xpstart}
4765 \newhooktoggle@series{Xpstarteverytime}%
4766
4767 \newhooktoggle@series{Xstanza}%
4768 \newhookcommand@series{Xstanzaseparator}%
4769
4770 \newhooktoggle@series{Xonlypstart}
4771 \newhooktoggle@series{Xnonbreakableafternumber}
4772 \newhooktoggle@series{Xlemmadisablefontselection}
4773 \newhookcommand@series@reload{Xmaxhnotes}{critical}
4774 \newhookcommand@series@reload{Xbeforenotes}{critical}
4775 \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
4776 \newhookcommand@series@reload{Xnotefontsize}{critical}
4777
4778 \newhookcommand@series{Xboxlinenum}%
4779 \newhookcommand@series{Xboxlinenumalign}%
4780
4781 \newhookcommand@series{Xboxstartlinenum}%
4782 \newhookcommand@series{Xboxendlinenum}%
4783
4784 \newhookcommand@series{Xafternote}%
4785 \newhookcommand@series{Xparafootsep}
4786
4787 \newhookcommand@series@reload{Xhsize}{critical}%
4788

```

```

4789 \fi
4790 \newhooktoggle@series{Xlineflag}[appref,SEref]
4791 \newhookcommand@series{Xtwolines}[appref,SEref]
4792 \newhookcommand@series{Xmorethantwolines}[appref,SEref]
4793 \newhookcommand@series{Xsublinesep}[appref,SEref,side]
4794 \newhooktoggle@series{Xtwolinesbutnotmore}[appref,SEref]
4795 \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref,SEref]
4796 \newhookcommand@series{Xlinerrangeseparator}[appref,SEref]
4797 %

```

### XXI.3.5 Options for familiar notes

Before generating the optional commands for familiar notes, we check the `\nofamiliar` option.

```

4798 \unless\ifnofamiliar@
4799   \newhooktoggle@series{parindentX}
4800   \newhookcommand@series{hangindentX}
4801   \newhookcommand@series{raggedX}
4802   \newhookcommand@series{hsize twocolX}
4803   \newhookcommand@series{hsize threecolX}
4804   \newhookcommand@series{colalignX}%
4805   \newhookcommand@series{notenumfontX}
4806   \newhookcommand@series{bhooknoteX}
4807   \newhookcommand@series@reload{bhookgroupX}{familiar}
4808   \newhookcommand@series@reload{beforenotesX}{familiar}
4809   \newhookcommand@series@reload{maxhnotesX}{familiar}
4810   \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
4811   \newhookcommand@series@reload{afterruleX}{familiar}
4812   \newhookcommand@series@reload{notefontsizeX}{familiar}
4813   \newhookcommand@series{afternoteX}
4814   \newhookcommand@series{parafootsepX}
4815   \newhookcommand@series@reload{hsizeX}{familiar}%
4816 \fi
4817 %

```

### 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.

```

4818 \unless\ifnoend@
4819   \newhookcommand@series{Xendnotenumfont}
4820   \newhookcommand@series{Xendlemmafont}%
4821   \newhookcommand@series{Xendbhooknote}
4822
4823   \newhookcommand@series{Xendboxlinenum}%
4824   \newhookcommand@series{Xendboxlinenumalign}%
4825

```

```

4826 \newhookcommand@series{Xendboxstartlinenum}%
4827 \newhookcommand@series{Xendboxendlinenum}%
4828
4829 \newhookcommand@series{Xendnotefontsize}
4830 \newhooktoggle@series{Xendlemmadisablefontselection}
4831 \newhookcommand@series{Xendlemmaseparator}
4832 \newhookcommand@series{Xendbeforelemmaseparator}
4833 \newhookcommand@series{Xendafterlemmaseparator}
4834 \newhookcommand@series{Xendinplaceoflemmaseparator}
4835
4836 \newhookcommand@series{Xendbeforenumber}%
4837 \newhookcommand@series{Xendafternumber}%
4838
4839 \newhooktoggle@series{Xendparagraph}
4840 \newhookcommand@series{Xendafternote}
4841 \newhookcommand@series{Xendsep}
4842
4843 \newhookcommand@series{Xendinplaceofnumber}%
4844 \newhooktoggle@series{Xendnonumber}%
4845
4846 \newhooktoggle@series{Xendnumberonlyfirstinline}%
4847 \newhooktoggle@series{Xendnumberonlyfirstintwolines}%
4848
4849 \newhookcommand@series{Xendsymmlinenumber}%
4850 \newhookcommand@series{Xendbeforesymmlinenumber}%
4851 \newhookcommand@series{Xendaftersymmlinenumber}%
4852 \newhookcommand@series{Xendboxsymmlinenumber}%
4853
4854 \newhookcommand@series{Xendbhooklinenumber}%
4855 \newhookcommand@series{Xendahooklinenumber}%
4856 \newhookcommand@series{Xendbhookinplaceofnumber}%
4857 \newhookcommand@series{Xendahookinplaceofnumber}%
4858
4859 \newhookcommand@series{Xendhangindent}%
4860
4861
4862 \fi
4863 \newhooktoggle@series{Xendlineflag}[apprefwithpage,SErefwithpage]
4864 \newhookcommand@series{Xendtwolines}[apprefwithpage,SErefwithpage]
4865 \newhookcommand@series{Xendmoreethantwolines}[apprefwithpage,SErefwithpage]
4866 \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage,SErefwithpage]
4867 \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage,
SErefwithpage]
4868 \newhookcommand@series{Xendlinerangeseparator}[apprefwithpage,SErefwithpage
]
4869 \newhookcommand@series{Xendbeforepagenumber}[apprefwithpage,SErefwithpage,
SErefonlypage]
4870 \newhookcommand@series{Xendafterpagenumber}[apprefwithpage,SErefwithpage]
4871 \newhookcommand@series{Xendlineprefixsingle}[apprefwithpage,SErefwithpage]
4872 \newhookcommand@series{Xendlineprefixmore}[apprefwithpage,SErefwithpage]

```

```

4873 \newhookcommand@series{Xendsublinesep}[apprefwithpage,Serefwithpage]
4874
4875 %

```

## XXI.4 Hooks for a particular footnote

`\newhooktoggle@specific` `\newhooktoggle@specific` is a generic command to create boolean hook specific to a note.

```

4876 \newcommand{\newhooktoggle@specific}[1]{%
4877   \newtoggle{#1}%
4878   \define@key[mac]{truefootnoteoption}{#1}[]{\global\settoggle{#1}{true}}%
   When enabling footnote option
4879   \define@key[mac]{falsefootnoteoption}{#1}[]{\global\settoggle{#1}{false}}
4880 }
4881 %

```

`\newhookarg@specific` `\newhookarg@specific` is a generic command to create argumen hook specific to a note.

```

4882 \newcommand{\newhookarg@specific}[1]{%
4883   \define@key[mac]{truefootnoteoption}{#1}{\global\def\linrangesep@{##1}}%
   When enabling footnote option
4884   \define@key[mac]{falsefootnoteoption}{#1}{\global\undef\linrangesep@}%
   When
4885 }
4886 %

```

And now, we define some hooks specific to a note.

```

4887 \newhooktoggle@specific{fulllines}%
4888 \newhooktoggle@specific{nonum}
4889 \newhooktoggle@specific{nosep}
4890 \newhookarg@specific{linrangesep}
4891 %

```

`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.

```

4892 %

```

`\nomk@` `\nomk@` toggle is used by `reledpar` to remove the footnote mark in the text when using `\footnoteXmk`. Read `reledpar` handbook.

```

4893 \newtoggle{nomk@}%
4894 %

```



## XXI.5 Alias

`\Xnolemmaseparator` `\Xnolemmaseparator[⟨series⟩]` is just an alias for `\Xlemmaseparator[⟨series⟩]{}`.

```
4895 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]{}}
4896 %
```

## 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`

```
4897 \countdef\pageno=0 \pageno=1
4898 \newcommand*{\advancepageno}{\ifnum\pageno<z@ \global\advance\pageno\m@ne
4899 \else\global\advance\pageno\@ne\fi}
4900
4901 %
```

### 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

`\doextrafeet` `\doextrafeet` 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.

```
4902 \newcommand*{\l@ddoxtrafeet}{%
4903 \IfStrEq{familiar-critical}{\@fnpos}
4904 {\do@feetX\Xdo@feet}%
4905 {%
4906 \IfStrEq{critical-familiar}{\@fnpos}%
4907 {\Xdo@feet\do@feetX}%
4908 {\do@feetX\Xdo@feet}%
4909 }%
4910 }%
4911
4912 %
```

`\Xdo@feet` `\Xdo@feet` is the code extending `\@makecol` to cater for the extra critical feet.

```

4913 \newcommand*\Xdo@feet}{%
4914   \setbox\@outputbox \vbox{%
4915     \unvbox\@outputbox
4916     \@opXfeet}}
4917 %

```

**\@opXfeet** The extra critical feet to be added to the output. The normal way to add one series, **\print@Xnotes** **\print@Xnotes**, is replaced by **reledpar** when using **\Pages**.

```

4918 \newcommand\print@Xnotes[1]{%
4919   \xdef\@currentseries{#1}%
4920   \csuse{#1footstart}{#1}%
4921   \csuse{#1footgroup}{#1}%
4922 }%
4923 %

```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```

4924 \newcommand*\@opXfeet}{%
4925   \unless\ifnocritical@%
4926     \gdef\firstXseries@{}%
4927     \def\do##1{%
4928       \ifvoid\csuse{##1footins}\else%
4929         \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4930         \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4931         \print@Xnotes{##1}%
4932       \fi%
4933     }%
4934     \dolistloop{\@series}%
4935   \fi%
4936 }%
4937 %

```

**\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**.

```

4938 \newcommand*\l@ddodoreintrafeet}{%
4939   \IfStrEq{familiar-critical}{\@fnpos}%
4940   {\@doreinfeetX\X@doreinfeet}%
4941   {%
4942     \IfStrEq{critical-familiar}{\@fnpos}%
4943     {\X@doreinfeet\@doreinfeetX}%
4944     {\@doreinfeetX\X@doreinfeet}%
4945   }%
4946 }
4947 %
4948 %

```

**\X@doreinfeet** **\X@doreinfeet** is the code for catering for the extra critical footnotes within **\@reinserts**.

```

4949 \newcommand*{\X@doreinfeet}{%
4950   \unless\ifnocritical%
4951   \def\do##1{%
4952     \ifvoid\csuse{##1footins}\else%
4953     \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4954     \fi}%
4955   \dolistloop{\@series}
4956   \fi%
4957 }
4958
4959 %

```

`\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
4960 \newcommand\print@notesX[1]{%
4961   \xdef\@currentseries{#1}%
4962   \csuse{footstart#1}{#1}%
4963   \csuse{footgroup#1}{#1}%
4964 }%
4965 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

4966 \newcommand*{\do@feetX}{%
4967   \unless\ifnofamiliar%
4968   \gdef\firstseriesX@{}%
4969   \setbox\@outputbox \vbox{%
4970     \unvbox\@outputbox%
4971     \def\do##1{%
4972       \ifvoid\csuse{footins##1}\else%
4973       \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
4974       \global\advance\skip\csuse{footins##1} by\csuse{afterruleX@##1}%
4975       \print@notesX{##1}%
4976       \fi%
4977     }%
4978     \dolistloop{\@series}}%
4979   \fi%
4980 }%
4981
4982 \newcommand{\@doreinfeetX}{%
4983   \unless\ifnofamiliar%
4984   \def\do##1{%
4985     \ifvoid\csuse{footins##1}\else
4986     \insert%
4987       \csuse{footins##1}
4988       {\unvbox\csuse{footins##1}}%
4989     \fi%
4990   }%
4991   \dolistloop{\@series}%

```

```

4992 \fi%
4993 }%
4994
4995 %

```

### 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.)

```

4996 \@ifclassloaded{memoir}{%
4997 %

```

memoir is loaded so we use memoir's built in hooks.

```

4998 \g@addto@macro{\m@mdoextrafeet}{\l@ddoxtrafeet}%
4999 \g@addto@macro{\m@mdodoreinextrafeet}{\l@ddodoreinxtrafeet}%
5000 }{%
5001 %

```

memoir has not been loaded, so patch `\@makecol` and `\@reinserts`.

```

5002 \@ifpackageloaded{fancyhdr}{%
5003 \patchcmd%
5004   {\latex@makecol}%
5005   {\xdef\@freelist{\@freelist\@midlist}}%
5006   {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5007   {}%
5008   {\led@error@fail@patch@makecol}%
5009 }{%
5010 \patchcmd%
5011   {\@makecol}%
5012   {\xdef\@freelist{\@freelist\@midlist}}%
5013   {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
5014   {}%
5015   {\led@error@fail@patch@makecol}%
5016 }%
5017
5018 \patchcmd%
5019   {\@reinserts}%
5020   {\ifvbox}%
5021   {\l@ddodoreinxtrafeet\ifvbox}%
5022   {}%
5023   {\led@error@fail@patch@reinserts}%
5024 }
5025
5026 %

```

It turns out that `\@doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```
5027 \newif\if@led@nofoot
5028
5029 %
```

```
5030 \@ifclassloaded{memoir}{%
5031 %
```

If the memoir class is loaded we hook into its modified `\@doclearpage`.

```
\@mem@extranofeet 5032 \g@addto@macro{\@mem@extranofeet}{%%
5033 \def\do#1{%
5034 \unless\ifnocritical@%
5035 \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
5036 \fi%
5037 \unless\ifnofamiliar@%
5038 \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
5039 \fi%
5040 }
5041 \dolistloop{\@series}%
5042 }%
5043 }{%
5044 %
```

As memoir is not loaded we have patch `\@doclearpage`.

```
\@led@testifnofoot 5045 \newcommand*{\@led@testifnofoot}{%
5046 \@doclearpage 5047 \@led@nofoottrue%
5048 \ifvoid\footins\else%
5049 \@led@nofootfalse%
5050 \fi%
5051 \def\do##1{%
5052 \unless\ifnocritical@%
5053 \ifvoid\csuse{##1footins}\else%
5054 \@led@nofootfalse%
5055 \fi%
5056 \unless\ifnofamiliar@%
5057 \ifvoid\csuse{footins##1}\else%
5058 \@led@nofootfalse%
5059 \fi%
5060 \fi%
5061 }%
5062 \dolistloop{\@series}%
5063 }%
5064
5065 \pretocmd%
5066 {\@doclearpage}%
```

```

5067 {\@led@testifnofoot}%
5068 {}%
5069 {\led@error@fail@patch@@doclearpage}%
5070
5071 \patchcmd%
5072 {\@doclearpage}%
5073 {\ifvoid\footins}%
5074 {\if@led@nofoot}%
5075 {}%
5076 {\led@error@fail@patch@@doclearpage}%
5077
5078 }
5079
5080 %

```

## 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.

```

5081 \list@create{\labelref@list}
5082 %

```

**\zz@@@** A convenience macro to zero two labeling counters in one go.

```

5083 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
5084
5085 %

```

**\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>32</sup>

<sup>32</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  $\TeX$  write methods for the .aux file.

Jesse Billett<sup>33</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.

```

5086 \newcommand*{\edlabel}[1]{%
5087   \ifl@dpairing\ifautopar%
5088     \strut%
5089   \fi\fi%
5090   \@bsphack%
5091   \ifbool{expr{bool{ledRcol} or bool{ledRcol@}}}{%
5092     \ifXnote@%
5093       \protected@write\@auxout{%
5094         {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1}}}%
5095       \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
5096     \else%
5097       \write\linenum@outR{\string\@lab}%
5098       \ifx\labelref@listR\empty%
5099         \xdef\label@refs{\zz@@@}%
5100       \else%
5101         \gl@p\labelref@listR\to\label@refs%
5102       \fi%
5103       \ifvmode%
5104         \advancelabel@refs%
5105       \fi%
5106   }%

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

5107   \protected@write\@auxout{%
5108     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR|{#1}}}%
5109   \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
5110   \fi%
5111 }{%
5112   \ifXnote@%
5113     \protected@write\@auxout{%
5114       {\string\l@dmake@labelsR\space\thepage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1}}}%
5115     \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
5116   \else%
5117     \write\linenum@out{\string\@lab}%
5118     \ifx\labelref@list\empty%
5119       \xdef\label@refs{\zz@@@}%
5120     \else%
5121       \gl@p\labelref@list\to\label@refs%
5122     \fi%

```

<sup>33</sup>(jdb43@cam.ac.uk) via the ctt thread ‘ledmac cross referencing’, 25 August 2003.

```

5123 \ifvmode%
5124 \advancelabel@refs%
5125 \fi%
5126 \protected@write\@auxout{%
5127 {\string\l@dmake@labels\space\thepage|\label@refs|\the\c@pstart
|{#1}}}%
5128 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}}}%
5129 \fi%
5130 }%
5131 \@esphack}%
5132
5133 %

```

`\advancelabel@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.

`\labelrefsparseline` 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 `\advancelabel@refs` command.

`\labelrefsparsesubline`

```

5134 \newcounter{line}%
5135 \newcounter{subline}%
5136 \newcommand{\advancelabel@refs}{%
5137 \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
5138 \stepcounter{line}%
5139 \ifsublines@
5140 \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
5141 %
5142 \stepcounter{subline}{1}%
5143 \def\label@refs{\theline|\thesubline}%
5144 \else%
5145 \def\label@refs{\theline|0}%
5146 \fi%
5147 }
5148 \def\labelrefsparseline#1|#2{#1}
5149 \def\labelrefsparsesubline#1|#2{#2}
5150 %

```

`\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.

```

5150 \newcommand*{\l@dmake@labels}{%
5151 \def\l@dmake@labels#1|#2|#3|#4|#5{%
5152 \expandafter\ifx\csname the@label#5\endcsname \relax\else
5153 \led@warn@DuplicateLabel{#5}%

```



```

5154 \fi
5155 \expandafter\gdef\csname the@label#5\endcsname{#1|#2|#3|#4|\relax}%
5156 \ignorespaces}
5157
5158 %

```

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.

```

5159 \AtBeginDocument{%
5160 \def\l@dmake@labels#1|#2|#3|#4|#5{%
5161 }
5162
5163 %

```

**\@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.

```

5164
5165 \newcommand*{\@lab}{%
5166 \ifledRcol
5167 \xright@appenditem{\linenumr@p{\line@numR}}{|%
5168 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
5169 \to\labelref@listR
5170 \else
5171 \xright@appenditem{\linenumr@p{\line@num}}{|%
5172 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
5173 \to\labelref@list
5174 \fi}
5175 %

```

**\applabel** \applabel, if called in \edtext will insert automatically both a start and an end label for the current edtext lines.

```

5176 \newcommand*{\applabel}[1]{%
5177 \ifnum\@edtext@level>0%
5178 %

```

Label should not be already defined.

```

5179 \ifcsundef{the@label#1}{%
5180 \csdef{the@label#1}{\applabel}%
5181 }%
5182 {%

```

```

5183     \led@warn@DuplicateLabel{#1 (applabel)}%
5184   }%
5185 %

```

Parse the `\edtext` line numbers.

```

5186     \expandafter\l@dp@rsefootspec\l@d@nums|%
5187 %

```

Use the  $\TeX$  standard hack for label.

```

5188     \@bsphack%
5189 %

```

And now, write the data in the auxiliary file.

```

5190     \ifledRcol%
5191       \protected@write\@auxout{}%
5192         {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
5193         \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
5194       \protected@write\@auxout{}%
5195         {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub|\the\c@pstartR|{#1:end}}}%
5196     \else%
5197       \protected@write\@auxout{}%
5198         {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstart|{#1:start}}%
5199         \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
5200       \protected@write\@auxout{}%
5201         {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub|\the\c@pstart|{#1:end}}}%
5202     \fi%
5203 %

```

Use the  $\TeX$  standard hack for label.

```

5204     \@esphack%
5205 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

5206     \else%
5207       \led@warn@AppLabelOutEdtext{#1}%
5208     \fi%
5209 %

```

End of `\applabel`

```

5210 }%
5211 %

```

`\edlabels` `\edlabels` and `\edlabelE` are just used to mark the beginning and the end of a passage.  
`\edlabelE`  
`\edlabelSE`

```

5212 \newcommand{\edlabelS}[1]{%
5213   \edlabel{#1:start}%
5214 }
5215 \newcommand{\edlabelE}[1]{%
5216   \edlabel{#1:end}%
5217 }
5218 \newcommand{\edlabelSE}[1]{%
5219   \edlabelS{#1}%
5220   \edlabelE{#1}%
5221 }
5222 %

```

**\wrap@edcrossref** `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

5223 \newrobustcmd{\wrap@edcrossref}[2]{%
5224   \ifdef{\hyperlink}%
5225     {\hyperlink{#1}{#2}}%
5226     {#2}%
5227 }
5228 %

```

**\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.

L<sup>A</sup>T<sub>E</sub>X already defines a `\pageref`, so changing the name to `\edpageref`.

```

5229 \newcommand*{\edpageref}[1]{\l@def@undefined{#1}\wrap@edcrossref{#1}{\
5230   l@getref@num{1}{#1}}}
5231 \newcommand*{\xpageref}[1]{\l@getref@num{1}{#1}}
5232 %

```

**\edlineref** If the specified label exists, `\lineref` gives its line number.

**\xlineref**

```

5233 \newcommand*{\edlineref}[1]{\l@def@undefined{#1}\wrap@edcrossref{#1}{\
5234   l@getref@num{2}{#1}\xflagref{#1}}}%
5235 \newcommand*{\xlineref}[1]{\l@getref@num{2}{#1}}%
5236 %

```

**\sublineref** If the specified label exists, `\sublineref` gives its sub-line number.

**\xsublineref**

```

5237 \newcommand*{\sublineref}[1]{\l@def@undefined{#1}\wrap@edcrossref{#1}{\
5238   l@getref@num{3}{#1}}}

```

```

5238 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}
5239
5240 %

```

**\pstartref** If the specified label exists, \pstartref gives its pstart number.

```

\pstartref
5241 \newcommand*\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{4}{#1}}}
5242 \newcommand*\xpstartref}[1]{\l@dgetref@num{4}{#1}}
5243
5244 %

```

**\xflagref** \xflagref finds the side flag of any ref defined with \edlabel.

```

5245 \newcommand*\xflagref}[1]{\l@dgetref@num{5}{#1}}
5246 %

```

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.

```

5247 \newcommand*\l@dref@undefined}[1]{%
5248 \expandafter\ifx\csname the@label#1\endcsname\relax
5249 \led@warn@RefUndefined{#1}%
5250 \fi}
5251
5252 %

```

**\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.

```

5253 \newcommand*\l@dgetref@num}[2]{%
5254 \expandafter
5255 \ifx\csname the@label#2\endcsname \relax
5256 000%
5257 \else
5258 \expandafter\expandafter\expandafter
5259 \l@dlabel@parse\csname the@label#2\endcsname|#1%
5260 \fi}
5261
5262 %

```

`\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`.)

```

5263 \newcommand*{\l@dlabel@parse}{%
5264 \def\l@dlabel@parse#1|#2|#3|#4|#5|#6{%
5265   \ifcase #6%
5266   \or #1%
5267   \or #2%
5268   \or #3%
5269   \or #4%
5270   \or #5%
5271   \fi}
5272 %

```

`\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.

```

5273 \newcommand*{\xxref}[2]{%
5274   {%
5275     \expandafter\ifx\csname the@label#1\endcsname \relax%
5276     \expandafter\let\csname the@@label#1\endcsname\zz@@@%
5277     \else%
5278       \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
5279 {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
5280       \fi%
5281       \expandafter\ifx\csname the@label#2\endcsname \relax%
5282       \expandafter\let\csname the@@label#2\endcsname\zz@@@%
5283       \else%
5284         \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
5285 {1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
5286         \fi%
5287         \letcs{\@tempa}{the@@label#1}%
5288         \letcs{\@tempb}{the@@label#2}%
5289         \linenum{\@tempa|
5290         \@tempb}}%
5291   }%

```

`\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).

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`.

```

5291 \def\Xtwolines@appref{}%
5292 \def\Xtwolines@SEref{}%
5293
5294 \def\Xmorethantwolines@appref{}%
5295 \def\Xmorethantwolines@SEref{}%
5296
5297 \def\Xlinerangeseparator@appref{\endashchar}%
5298 \def\Xlinerangeseparator@SEref{\endashchar}%
5299
5300 \def\Xsublinesep@appref{\fullstop}%
5301 \def\Xsublinesep@SEref{\fullstop}%
5302
5303 \newtoggle{Xtwolinesbutnotmore@appref}%
5304 \newtoggle{Xtwolinesbutnotmore@SEref}%
5305
5306 \newtoggle{Xtwolinesonlyinsamepage@appref}%
5307
5308 \newtoggle{Xtwolinesonlyinsamepage@SEref}%
5309
5310 \newtoggle{Xlineflag@appref}%
5311 \toggletrue{Xlineflag@appref}%Here exception
5312 \newtoggle{Xlineflag@SEref}%
5313 \toggletrue{Xlineflag@SEref}%%Here exception
5314
5315 \def\Xendtwolines@apprefwithpage{}%
5316 \def\Xendtwolines@SErefwithpage{}%
5317
5318 \def\Xendmorethantwolines@apprefwithpage{}%
5319 \def\Xendmorethantwolines@SErefwithpage{}%
5320
5321 \def\Xendlinerangeseparator@apprefwithpage{\endashchar}
5322 \def\Xendlinerangeseparator@SErefwithpage{\endashchar}
5323 \def\Xendlinerangeseparator@SErefonlypage{\endashchar}
5324
5325 \def\Xendbeforepagenumber@apprefwithpage{p.}%
5326 \def\Xendbeforepagenumber@SErefwithpage{p.}%
5327 \def\Xendbeforepagenumber@SEonlypage{p.}%
5328
5329 \def\Xendafterpagenumber@apprefwithpage{ }%
5330 \def\Xendafterpagenumber@SErefwithpage{ }%
5331
5332

```

```

5333 \def\Xendlineprefixsingle@apprefwithpage{}%
5334 \def\Xendlineprefixsingle@SErefwithpage{}%
5335
5336 \def\Xendlineprefixmore@apprefwithpage{}%
5337 \def\Xendlineprefixmore@SErefwithpage{}%
5338
5339 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
5340 \newtoggle{Xendtwolinesbutnotmore@SErefwithpage}%
5341
5342 \def\Xendsublinesep@apprefwithpage{\fullstop}%
5343 \def\Xendsublinesep@SErefwithpage{\fullstop}%
5344
5345 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
5346 \newtoggle{Xendtwolinesonlyinsamepage@SErefwithpage}%
5347
5348 \newtoggle{Xendlineflag@apprefwithpage}
5349 \toggletrue{Xendlineflag@apprefwithpage}%Here, exception
5350 \newtoggle{Xendlineflag@SErefwithpage}
5351 \toggletrue{Xendlineflag@SErefwithpage}%Here, exception
5352
5353 %

```

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.

```

5354
5355 \xdef\Xboxstartlinenum@appref{Opt}
5356 \xdef\Xboxstartlinenum@SEref{Opt}
5357
5358 \xdef\Xboxendlinenum@appref{Opt}
5359 \xdef\Xboxendlinenum@SEref{Opt}
5360
5361 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
5362 \xdef\Xendboxstartlinenum@SErefwithpage{Opt}
5363
5364 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
5365 \xdef\Xendboxendlinenum@SErefwithpage{Opt}
5366
5367 %

```

Now, declare the default values of \@apprefprefixsingle and \@apprefprefixmore, \@SErefprefix, \@SErefprefixmore and the commands which defines them.

```

5368 \newcommand\@apprefprefixsingle{}%
5369 \newcommand\@SErefprefixsingle{}%
5370
5371 \newcommand\@apprefprefixmore{}%
5372 \newcommand\@SErefprefixmore{}%
5373
5374 \newcommand{\setapprefprefixsingle}[1]{%

```

```

5375 \gdef\@apprefprefixsingle{#1}%
5376 }
5377 \newcommand{\setSErefprefixsingle}[1]{%
5378 \gdef\@SErefprefixsingle{#1}%
5379 }
5380
5381 \newcommand{\setapprefprefixmore}[1]{%
5382 \gdef\@apprefprefixmore{#1}%
5383 }
5384 \newcommand{\setSErefprefixmore}[1]{%
5385 \gdef\@SErefprefixmore{#1}%
5386 }
5387
5388 %

```

And not `\setSErefonlypageprefixsingle` and `\setSErefonlypageprefixmore`.

```

5389 \let\setSErefonlypageprefixsingle\XendbeforepagenumberSErefonlypage%
5390 \newcommand{\setSErefonlypageprefixmore}[1]{%
5391 \gdef\SErefonlypage@prefixmore{#1}%
5392 }%
5393 %

```

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.

```

5394
5395 \newcommandx{\appref}[2][1,usedefault]{\reformatted@{#1}{#2}{appref}}
5396 \newcommandx{\SEref}[2][1,usedefault]{\reformatted@{#1}{#2}{SEref}}
5397
5398 \newcommandx{\apprefwithpage}[2][1,usedefault]{\reformattedwithpage@
5399 {#1}{#2}{appref}}
5400 \newcommandx{\SErefwithpage}[2][1,usedefault]{\reformattedwithpage@
5401 {#1}{#2}{SEref}}
5402
5403 \newcommand{\reformatted@}[3]{%
5404 \def\do##1{%
5405 \setkeys[mac]{truefootnoteoption}{##1}%
5406 }%
5407 \notblank{#1}{\docsvlist{#1}}}%
5408 \xdef\@currentseries{#3}%
5409 \ifcempty{@#3prefixmore}%
5410 {\@apprefprefixsingle}%
5411 {%
5412 \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
5413 {\csuse{@#3prefixsingle}}%

```



```

5414     {\csuse{@#3prefixmore}}}%
5415   }%
5416   \ifboolexpr{%
5417     test{\ifcsundef{the@label#2:start}}}%
5418     or test{\ifcsundef{the@label#2:end}}}%
5419   }%
5420   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}}%
5421   {%
5422     \def\@this@crossref@start{#2:start}%
5423     \def\@this@crossref@end{#2:end}%
5424     \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|\%
5425     \undef\@this@crossref@end%
5426     \undef\@this@crossref@start%
5427   }%
5428   \def\do##1{%
5429     \setkeys[mac]{falsefootnoteoption}{##1}%
5430   }%
5431   \notblank{#1}{\docsvlist{#1}}{ }%
5432 }%
5433
5434 \newcommand{\reformattedwithpage@}[3]{%
5435   \def\do##1{%
5436     \setkeys[mac]{truefootnoteoption}{##1}%
5437   }%
5438   \notblank{#1}{\docsvlist{#1}}{ }%
5439   \xdef\@currentseries{#3withpage}%
5440   \ifboolexpr{%
5441     test{\ifcsundef{the@label#2:start}}}%
5442     or test{\ifcsundef{the@label#2:end}}}%
5443   }%
5444   {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}}%
5445   {%
5446     \def\@this@crossref@start{#2:start}%
5447     \def\@this@crossref@end{#2:end}%
5448     \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}|\relax|\
xflagref{#2:start}|\%
5449     \undef\@this@crossref@end%
5450     \undef\@this@crossref@start%
5451   }%
5452   \def\do##1{%
5453     \setkeys[mac]{falsefootnoteoption}{##1}%
5454   }%
5455   \notblank{#1}{\docsvlist{#1}}{ }%
5456 }%
5457
5458 \newcommand{\reformattedonlypage@}[3]{%
5459   \def\do##1{%

```

```

5460 \setkeys[mac]{truefootnoteoption}{##1}%
5461 }%
5462 \notblank{#1}{\docsvlist{#1}}{}%
5463 \xdef\@currentseries{#3onlypage}%
5464 \ifboolexpr{%
5465   test{\ifcsundef{the@label#2:start}}}%
5466   or test{\ifcsundef{the@label#2:end}}}%
5467 }%
5468 {\led@warn@pairRefUndefined{#2}\nfss@text{\reset@font\bfseries ??}}%
5469 {\ifnumequal{\xpageref{#2:end}}{\xpageref{#2:start}}}%
5470   {%
5471     \printnpnum{%
5472       \wrap@edcrossref{#2:start}{\xpageref{#2:start}}}%
5473     }%
5474   }%
5475 {%
5476   \ifcsvoid{#3onlypage@prefixmore}%
5477     {}%
5478     {\csletcs{Xendbeforepagenumber@#3onlypage}{#3onlypage@prefixmore}}%
5479   \ifdefined\linrangesep@%
5480     \printnpnum{%
5481       \wrap@edcrossref{#2:start}{\xpageref{#2:start}}}%
5482     \linrangesep@%
5483     \wrap@edcrossref{#2:end}{\xpageref{#2:end}}}%
5484   }%
5485   \else%
5486     \printnpnum{%
5487       \wrap@edcrossref{#2:start}{\xpageref{#2:start}}}%
5488     \csuse{Xendlinrangeseparator@\@currentseries}%
5489     \wrap@edcrossref{#2:end}{\xpageref{#2:end}}}%
5490   }%
5491   \fi%
5492 }%
5493 }%
5494 \def\do##1{%
5495   \setkeys[mac]{falsefootnoteoption}{##1}%
5496   }%
5497 \notblank{#1}{\docsvlist{#1}}{}%
5498 }%
5499 %

```

`\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`.

```

5500 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\

```

```

endcsname{#2}}
5501
5502 %

```

(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. 116 and V.9 p. 86), since `\xxref` makes a call to `\linenum` in order to do its work.)

## XXIV Side notes

Regular `\marginpar`s 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`s in numbered text do not disturb the flow.

```

5503 \pretocmd{\@xympar}%
5504   {\ifnumberedpar@
5505     \led@warn@NoMarginpars
5506     \@esphack
5507   \else}%
5508   {}%
5509   {}%
5510
5511 \apptocmd{\@xympar}%
5512   {\fi}%
5513   {}
5514   {}
5515
5516 %

```

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

```

5517 \newcount\sidenote@margin
5518 \newcommand*{\sidenotemargin}[1]{\{%
5519   \l@dgetsidenote@margin{#1}%

```

```

5520 \ifnum\@l@dttempcntb>\m@ne
5521 \ifledRcol
5522 \global\sidenote@marginR=\@l@dttempcntb
5523 \else
5524 \global\sidenote@margin=\@l@dttempcntb
5525 \fi
5526 \fi}}
5527 \newcommand*{\l@dgetsidenote@margin}[1]{%
5528 \def\@tempa{#1}\def\@tempb{left}%
5529 \ifx\@tempa\@tempb
5530 \l@dttempcntb \z@
5531 \else
5532 \def\@tempb{right}%
5533 \ifx\@tempa\@tempb
5534 \l@dttempcntb \@ne
5535 \else
5536 \def\@tempb{outer}%
5537 \ifx\@tempa\@tempb
5538 \l@dttempcntb \tw@
5539 \else
5540 \def\@tempb{inner}%
5541 \ifx\@tempa\@tempb
5542 \l@dttempcntb \thr@@
5543 \else
5544 \led@warn@BadSidenotemargin
5545 \l@dttempcntb \m@ne
5546 \fi
5547 \fi
5548 \fi
5549 \fi}
5550 \sidenotemargin{right}
5551
5552 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
5553 \newbox\l@dlp@rbox
5554 \newbox\l@drp@rbox
5555
5556 %

```

`\ledlsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`), their distance from the text (initialised to `\linenumsep`), and the fonts used.

```

\ledrsnotewidth
\ledlsnotesep
5557 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
5558 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup
5559 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup
5560 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
5561 \newcommand*{\ledlsnotefontsetup}{\raggedleft\footnotesize}
5562 \newcommand*{\ledrsnotefontsetup}{\raggedright\footnotesize}

```

```

5563
5564 %

\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
5565 \newcommand*{\ledleftnote}[1]{\edtext{}{\l@dlsnote{#1}}}
5566 \newcommand*{\ledrightnote}[1]{\edtext{}{\l@drsnote{#1}}}
5567
5568 \newcommand*{\ledinnernote}[1]{%
5569 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5570 \ledleftnote{#1}%
5571 \else%
5572 \ledrightnote{#1}%
5573 \fi%
5574 }
5575
5576 \newcommand*{\ledouternote}[1]{%
5577 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5578 \ledrightnote{#1}%
5579 \else%
5580 \ledleftnote{#1}%
5581 \fi%
5582 }
5583
5584 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcnote{#1}}}
5585 %

```

**\l@dlsnote** . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

**\l@drsnote**

**\l@dcnote**

```

5586 \newif\ifrightnoteup
5587 \rightnoteuptrue
5588
5589 \newcommand*{\l@dlsnote}[1]{%
5590 \begingroup%
5591 \newcommand{\content}{#1}%
5592 \ifnumberedpar@
5593 \ifledRcol%
5594 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
5595 \to\inserts@listR
5596 \global\advance\insert@countR \c@ne%
5597 \else%
5598 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
5599 \to\inserts@list
5600 \global\advance\insert@count \c@ne%

```

```

5601 \fi
5602 \fi\ignorespaces\endgroup}
5603
5604 \newcommand*{\l@drsnote}[1]{%
5605 \begingroup%
5606 \newcommand{\content}{#1}%
5607 \ifnumberedpar@
5608 \ifledRcol%
5609 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}{%
5610 \to\inserts@listR
5611 \global\advance\insert@countR \@ne%
5612 \else%
5613 \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}{%
5614 \to\inserts@list
5615 \global\advance\insert@count \@ne%
5616 \fi
5617 \fi\ignorespaces\endgroup}
5618
5619 \newcommand*{\l@dcnote}[1]{%
5620 \begingroup%
5621 \newcommand{\content}{#1}%
5622 \ifnumberedpar@
5623 \ifledRcol%
5624 \xright@appenditem{\noexpand\vl@dcnote{\expandonce\content}}{%
5625 \to\inserts@listR
5626 \global\advance\insert@countR \@ne%
5627 \else%
5628 \xright@appenditem{\noexpand\vl@dcnote{\expandonce\content}}{%
5629 \to\inserts@list
5630 \global\advance\insert@count \@ne%
5631 \fi
5632 \fi\ignorespaces\endgroup}
5633
5634 %

```

**\vl@dlsnote** Put the left/right text into boxes, but just save the moveable text. **\l@dcnotetext**, **\vl@drsnote** **\l@dcnotetext@l** and **\l@dcnotetext@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@dcnotetext** in any cases.
- Store the content of **\rightsidenote** to:
  - **\l@dcnotetext** if **\ledsidenote** is to be put on right.
  - **\l@dcnotetext@r** if **\ledsidenote** is to be put on left.
- Store the content of **\leftsidenote** to:

- \l@dcstetext if \ledsidenote is to be put on left.
- \l@dcstetext@l if \ledsidenote is to be put on right.

```

5635 \newcommand*{\vl@dlsnote}[1]{%
5636   \ifledRcol{%
5637     \@l@tempcntb=\sidenote@marginR%
5638     \ifnum\@l@tempcntb>\@ne%
5639       \advance\@l@tempcntb by\page@numR%
5640     \fi%
5641   \else%
5642     \@l@tempcntb=\sidenote@margin%
5643     \ifnum\@l@tempcntb>\@ne%
5644       \advance\@l@tempcntb by\page@num%
5645     \fi%
5646   \fi%
5647   \ifodd\@l@tempcntb%
5648     \listgadd{\l@dcstetext@l}{#1}%
5649   \else%
5650     \listgadd{\l@dcstetext}{#1}%
5651   \fi
5652 }
5653 \newcommand*{\vl@drsnote}[1]{%
5654   \ifledRcol{%
5655     \@l@tempcntb=\sidenote@marginR%
5656     \ifnum\@l@tempcntb>\@ne%
5657       \advance\@l@tempcntb by\page@numR%
5658     \fi%
5659   \else%
5660     \@l@tempcntb=\sidenote@margin%
5661     \ifnum\@l@tempcntb>\@ne%
5662       \advance\@l@tempcntb by\page@num%
5663     \fi%
5664   \fi%
5665   \ifodd\@l@tempcntb%
5666     \listgadd{\l@dcstetext}{#1}%
5667   \else%
5668     \listgadd{\l@dcstetext@r}{#1}%
5669   \fi%
5670 }
5671 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcstetext}{#1}}
5672
5673 %

```

`\setl@dlp@rbox` `\setl@dlprbox{<lednums>}{<tag>}{<text>}` puts <text> into the \l@dlp@rbox box. And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```

5674 \newcommand*{\setl@dlp@rbox}[1]{%
5675   {\parindent\z@\hspace=\ledlsnotewidth\ledlsnotefontsetup

```

```

5676 \global\setbox\l@dlp@rbox
5677 \ifleftnoteup
5678   =\vbox to\z@{\vss #1}%
5679 \else
5680   =\vbox to 0.70\baselineskip{\strut#1\vss}%
5681 \fi}}
5682 \newcommand*\setl@drp@rbox}[1]{%
5683 {\parindent\z@\hspace=\ledrsnotewidth\ledrsnotefontsetup
5684 \global\setbox\l@drp@rbox
5685 \ifrightnoteup
5686   =\vbox to\z@{\vss#1}%
5687 \else
5688   =\vbox to0.7\baselineskip{\strut#1\vss}%
5689 \fi}}
5690 \newif\ifleftnoteup
5691 \leftnoteuptrue
5692 %

```

**\@sidenotesep** This macro is used to separate sidenotes of the same line.

```

5693 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
5694 \newcommand{\@sidenotesep}{, }
5695 %

```

**\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 `\@sidenotesep` as separator. It is the result that we put on the sidenote.

```

5696 \newcommand*\affixside@note}{%
5697   \def\sidenotecontent@{}%
5698   \numgdef\itemcount@{0}%
5699   \def\do##1{%
5700     \ifnumequal{\itemcount@}{0}%
5701     {%
5702       \appto\sidenotecontent@{##1}}% Not print not separator before
5703       the 1st note
5704     }%
5705     \numgdef\itemcount@{\itemcount@+1}%
5706   }%
5707   \dolistloop{\l@dcnsnotetext}%
5708   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5709 %

```

And we do the same for left and right notes (not movable).

```

5710 \gdef\@templ@d{}%
5711 \gdef\@templ@n{\l@dcnsnotetext\l@dcnsnotetext@1\l@dcnsnotetext@r}%

```



```

5712 \ifx\@templ@d\@templ@n \else%
5713 \if@twocolumn%
5714 \if@firstcolumn%
5715 \setl@dlp@rbox{##1}{\sidenotecontent@}%
5716 \else%
5717 \setl@drp@rbox{\sidenotecontent@}%
5718 \fi%
5719 \else%
5720 \l@dttempcntb=\sidenote@margin%
5721 \ifnum\l@dttempcntb>\@ne%
5722 \advance\l@dttempcntb by\page@num%
5723 \fi%
5724 \ifodd\l@dttempcntb%
5725 \setl@drp@rbox{\sidenotecontent@}%
5726 \gdef\sidenotecontent@{}%
5727 \numgdef{\itemcount@}{0}%
5728 \dolistloop{\l@dcsnotetext@l}%
5729 \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5730 \setl@dlp@rbox{\sidenotecontent@}%
5731 \else%
5732 \setl@dlp@rbox{\sidenotecontent@}%
5733 \gdef\sidenotecontent@{}%
5734 \numgdef{\itemcount@}{0}%
5735 \dolistloop{\l@dcsnotetext@r}%
5736 \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5737 \setl@drp@rbox{\sidenotecontent@}%
5738 \fi%
5739 \fi%
5740 \fi%
5741 }
5742 %

```

## 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@dfeetbeginmini` These will be the hooks in `\@iiiminipage` and `\endminipage`.  
`\l@dfeetendmini` They can be extended to handle other things if necessary.

```

5743 \ifnoledgroup@ \else%
5744 \newcommand*{\l@dfeetbeginmini}{\@ledgrouptrue\l@dedbeginmini\l@dfambeginmini}
5745 \newcommand*{\l@dfeetendmini}{%
5746 \IfStrEq{critical-familiar}{\@mpfnpos}%
5747 {\l@dedendmini\l@dfamendmini}%
5748 {%

```

```

5749         \IfStrEq{familiar-critical}{\@mpfnpos}%
5750         {\l@dfamendmini\l@dedendmini}%
5751         {\l@dedendmini\l@dfamendmini}%
5752     }%
5753 }%
5754 %

```

`\l@dedbeginmini` These handle the initiation and closure of critical footnotes in a minipage environment.

```

\l@dedendmini
5755 \newcommand*\l@dedbeginmini{%
5756     \unless\ifnocritical@%
5757     \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5758     \dolistloop{\@series}%
5759     \fi%
5760 }
5761 \newcommand*\l@dedendmini{%
5762     \unless\ifnocritical@%
5763     \ifl@dpairing%
5764         \ifledRcol%
5765             \flush@notesR%
5766         \else%
5767             \flush@notes%
5768         \fi%
5769     \fi
5770     \def\do##1{%
5771         \ifvoid\csuse{mp##1footins}\else%
5772             \ifl@dpairing\ifparledgroup%
5773                 \ifledRcol%
5774                     \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@nameuse{mp##1footins}}%
5775                 \else%
5776                     \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
+skip\@nameuse{mp##1footins}}%
5777                 \fi%
5778             \fi\fi%
5779             \csuse{mp##1footgroup}{##1}%
5780         \fi}%
5781     \dolistloop{\@series}%
5782     \fi%
5783 }%
5784 %
5785 %

```

`\l@dfambeginmini` These handle the initiation and closure of familiar footnotes in a minipage environment.

```

\l@dfamendmini
5786 \newcommand*\l@dfambeginmini{%
5787     \unless\ifnofamiliar@%
5788     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5789     \dolistloop{\@series}%
5790     \fi%

```

```

5791 }%
5792
5793 \newcommand*{\l@dfamendmini}{%
5794 \unless\ifnofamiliar%
5795 \def\do##1{%
5796 \ifvoid\csuse{mpfootins##1}\else%
5797 \csuse{mpfootgroup##1}{##1}%
5798 \fi}%
5799 \dolistloop{\@series}%
5800 \fi%
5801 }%
5802 %

```

`\@iiminipage` This is our extended form of the kernel `\@iiminipage` defined in `ltboxes.dtx`.

```

5803 \patchcmd%
5804 {\@iiminipage}%
5805 {\let\@footnotetext\@mpfootnotetext}%
5806 {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
5807 {}%
5808 {\led@error@fail@patch@\@iiminipage}%
5809 %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5810 \patchcmd%
5811 {\endminipage}%
5812 {\footnoterule}%
5813 {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5814 {}%
5815 {\led@error@fail@patch@endminipage}%
5816
5817 \patchcmd%
5818 {\endminipage}%
5819 {\@minipagefalse}%
5820 {\l@dfeetendmini\@minipagefalse}%
5821 {}%
5822 {\led@error@fail@patch@endminipage}%
5823
5824 %

```

`\l@dunboxmpfoot` `\@ldunboxmpfoot` insert normal footnotes for `ledgroup`.  
`edgroup@beforenormalnotes`

```

5825 \newcommand*{\l@dunboxmpfoot}{%
5826 \vskip\skip\@mpfootins
5827 \normalcolor
5828 \footnoterule
5829 \l@advance@parledgroup@beforenormalnotes
5830 \unvbox\@mpfootins%
5831 }
5832 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5833 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5834   \ifparledgroup
5835     \ifl@pairing
5836       \ifledRcol
5837         \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@mpfootins}
5838       \else
5839         \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+
skip\@mpfootins}
5840       \fi
5841     \fi
5842   \fi
5843 }
5844 %

```

**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.

```

5845
5846 \newenvironment{ledgroup}{%
5847   \resetprevpage@num%
5848   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5849   \let\@footnotetext\@mpfootnotetext
5850   \l@dfeetbeginmini%
5851 }{%
5852   \par
5853   \unskip
5854   \ifvoid\@mpfootins\else
5855     \l@dunboxmpfoot
5856   \fi
5857   \l@dfeetendmini%
5858   \@ledgroupfalse%
5859 }
5860
5861
5862 %

```

**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.

```

5863 \newenvironment{ledgroupsize}[2][1]{%
5864 %

```

Set the various text measures.

```

5865 \hsize #2\relax
5866 %

```

Initialize fills for centering.

```

5867 \let\ledllfill\hfil
5868 \let\ledrlfill\hfil
5869 \def\@tempa{#1}\def\@tempb{1}%
5870 %

```

Left adjusted numbered lines

```

5871 \ifx\@tempa\@tempb
5872 \let\ledllfill\relax
5873 \else
5874 \def\@tempb{r}%
5875 \ifx\@tempa\@tempb
5876 %

```

Right adjusted numbered lines

```

5877 \let\ledrlfill\relax
5878 \fi
5879 \fi
5880 %

```

Set up the footnoting.

```

5881 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5882 \let\@footnotetext\@mpfootnotetext
5883 \l@dfetbeginmini%
5884 }{
5885 \par
5886 \unskip
5887 \ifvoid\@mpfootins\else
5888 \l@dunboxmpfoot
5889 \fi
5890 \l@dfetendmini%
5891 }
5892 %
5893 %

```

Close the \ifnoledgroup@else.

```

5894 \fi%
5895 %

```

**\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.

```

5896 \newif\ifledgroupnotesL@
5897 \newif\ifledgroupnotesR@
5898 %

```

## 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`.

```

5899 \AtBeginDocument{%
5900   \unless\ifl@imakeidx%
5901     \ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5902   \fi%
5903   \unless\ifl@indextools%
5904     \ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5905   \fi%
5906 }
5907 %

```

### 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
5908 \newcommand{\pagelinesep}{-}
5909 \newcommand{\edindexlab}{$&}
5910 \newcounter{labidx}
5911 \setcounter{labidx}{0}
5912
5913 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5914 \newcommand{\doedindexlabel}{%
5915   \stepcounter{labidx}%
5916   \edlabel{\edindexlab\thelabidx}%
5917 }
5918
5919 %

```

`\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`.

```

5920 \newcounter{pageline}%
5921 \renewcommand{\thepageline}{%
5922   \thepage%
5923   \pagelinesep%
5924   \xlineref{\edindexlab\thelabidx}%
5925 }
5926 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command is called in critical notes.  
`\theendpageline`

```

5927 \newcommand{\thestartpageline}{%
5928   \l@dparsedstartpage%
5929   \pagelinesep%
5930   \l@dparsedstartline%
5931 }
5932 \newcommand{\theendpageline}{%
5933   \l@dparsedendpage%
5934   \pagelinesep%
5935   \l@dparsedendline%
5936 }
5937 %

```

### 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.

```

5938 \newif\if@edindex@fornote@
5939 %

```

`\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 `|`.

```

5940 \newcommand{\prepare@edindex@fornote}[1]{%
5941   \l@dp@rsefootspec#1|%
5942   \@edindex@fornote@true%
5943 }
5944 %

```

`\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.

```

5945 \newcommand{\get@edindex@ledinnote@command}{%
5946   \ifxindy@%
5947     \gdef\@ledinnote@command{%
5948       ledinnote\thelabidx%
5949     }%
5950     \ifxindyhyperref@%
5951       \immediate\write\eledmac@xindy@out{%
5952         (define-attributes ("ledinnote\thelabidx"))^^J
5953         \space\space(markup-locref^^J
5954         \eledmacmarkuplocrefdepth^^J

```

```

5955 :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
    }{"^^J
5956 :close "}"^^J
5957 :attr "ledinnote\thelabidx"^^J
5958 )
5959 }%
5960 \else%
5961 \immediate\write\eledmac@xindy@out{%
5962 (define-attributes ("ledinnote\thelabidx"))^^J
5963 \space\space(markup-locref^^J
5964 \eledmacmarkuplocdepth^^J
5965 :open "\string\ledinnote{\@index@command}{"^^J
5966 :close "}"^^J
5967 :attr "ledinnote\thelabidx"^^J
5968 )
5969 }%
5970 \fi%
5971 %

```

If we do not use xindy option, `\@ledinnote@command` will produce something like `ledinnote{formattingcommand}`.

```

5972 \else%
5973 \gdef\@ledinnote@command{%
5974 ledinnote[\edindexlab\thelabidx]{\@index@command}%
5975 }%
5976 \fi%
5977 }
5978 %

```

## 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 |.

```

5979 \def\get@index@command#1|#2+{%
5980 \gdef\@index@txt{#1}%
5981 \gdef\@index@command{#2}%
5982 \xdef\@index@parenthesis{}%
5983 \IfBeginWith{\@index@command}{(}{%
5984 \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5985 \global\let\@index@command\@index@command@%
5986 \xdef\@index@parenthesis{(%}%
5987 }{}%
5988 \IfBeginWith{\@index@command}{)}{%
5989 \StrGobbleLeft{\@index@command}{1}[\@index@command@]%
5990 \global\let\@index@command\@index@command@%
5991 \xdef\@index@parenthesis{)}%
5992 }{}%
5993 }
5994 %

```



## 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.

`\ledinnotehyperpage`

`\ledinnotemark`

```

5995 \newcommandx{\ledinnote}[3][1,usedefault]{%
5996   \ifboolexpr{%
5997     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
5998     or%
5999     bool {xindyhyperref@}%
6000   }%
6001   {%
6002     \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
6003   }%
6004   {%
6005     \csuse{#2}{\ledinnotemark{#3}}%
6006   }%
6007 }%
6008 \newcommand{\ledinnotehyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
6009   {#2}}}%
6010 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
6011 %

```

## 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`

```

6011 \newcommandx{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
6012   index name, #2 = the text
6013   \ifl@imakeidx%
6014     \if@edindex@fornote%
6015     \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
6016     \get@edindex@ledinnote@command%
6017     \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
6018   }{\thestartpageline}%
6019     \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
6020   }{\theendpageline}%
6021   \else%
6022     \get@edindex@hyperref{#2}%
6023     \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%

```

```

6021 \fi%
6022 \else%
6023 \if@edindex@fornote@%
6024 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
6025 \get@edindex@ledinnote@command%
6026 \expandafter\protected@write\@indexfile{%
6027 {\string\indexentry{\@index@txt|(\@ledinnote@command){\thestartpageline}
6028 }%
6029 \expandafter\protected@write\@indexfile{%
6030 {\string\indexentry{\@index@txt|)\@ledinnote@command){\theendpageline}
6031 }%
6032 \else%
6033 \protected@write\@indexfile{%
6034 {\string\indexentry{#2){\thepageline}
6035 }%
6036 \fi%
6037 \fi%
6038 \endgroup
6039 \@esphack%
6040 }
6041 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

6042 \pretocmd{\makeindex}{%
6043 \def\edindex{\@bsphack
6044 \doedindexlabel
6045 \begingroup
6046 \@sanitize
6047 \@wredindex}}{}{}
6048 \newcommand{\edindex}[1]{\@bsphack\@esphack}
6049 %

```

## XXVI.7 Hyperlink

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

6050 \newcommand{\hyperlinkformat}[3]{%
6051 \ifstrempy{#1}%
6052 {\hyperlink{#2}{#3}}%
6053 {\csuse{#1}{\hyperlink{#2}{#3}}%
6054 }}
6055 %

```

`\hyperlinkR` `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.

```

6056 \newcommand{\hyperlinkR}[2]{%

```

```

6057 \hyperlink{#1}{#2\@Rlineflag}%
6058 }%
6059
6060 %

```

**\hyperlinkformatR** \hyperlinkformatR command is to be used to create a internal hyperlink, a format and a \@Rlineflag, when indexing.

```

6061 \newcommand{\hyperlinkformatR}[3]{%
6062   \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
6063 }%
6064
6065 %

```

**\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).

```

6066 \newcommand{\get@edindex@hyperref}[1]{%
6067   %

```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```

6068 \edef\temp@{%
6069   \catcode`\ =9 %space need for catcode
6070   \detokenize{#1}%For active character in unicode
6071   \catcode`\ =10 % space need for catcode
6072 }%
6073 %

```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```

6074 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
6075   \IfSubStr{\temp@}{|}%
6076   {\get@index@command#1+%
6077   \ifledRcol%
6078     \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
6079     hyperlinkformatR{\@index@command}%
6080     {\edindexlab\thelabidx}}%
6081   \else%
6082     \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
6083     hyperlinkformat{\@index@command}%
6084     {\edindexlab\thelabidx}}%
6085   \fi%
6086 }%
6087 {\get@index@command#1|+%
6088   \ifledRcol%
6089     \gdef\@edindex@hyperref{|\hyperlinkR{\edindexlab\thelabidx}}%
6090   \else%
6091     \gdef\@edindex@hyperref{|\hyperlink{\edindexlab\thelabidx}}%
6092   \fi%
6093 }%
6094 }%
6095 %

```

```

6096 % If we use both xindy and hyperref, first get the \protect\cs{
      index@command} command.
6097 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
6098 % \begin{macrocode}
6099 {\ifxindyhyperref%
      \IfSubStr{\temp@}{|}%
6100      {\get@index@command#1+}%
6101      {\get@index@command#1|+}%
6102      \gdef\@edindex@hyperref{|eledmac\thelabidx}%
6103 %
6104 %

```

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.

```

6105 \IfStrEq{\@index@parenthesis}{(}%
6106 {%
6107 \csxdef{xindyparenthesis@\@index@txt}{\thelabidx}%
6108 \gdef\@edindex@hyperref{|(eledmac\thelabidx}%
6109 }%
6110 {}%
6111 %

```

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.

```

6112 \IfStrEq{\@index@parenthesis}{)}%
6113 {%
6114 \xdef\@edindex@hyperref{)|(eledmac\csuse{xindyparenthesis@\@index@txt}}%
6115 \global\csundef{xindyparenthesis@\@index@txt}%
6116 }%
6117 %

```

Write in the .xdy file the attributes of the location.

```

6118 {%
6119 \immediate\write\eledmac@xindy@out{%
6120 (define-attributes ("eledmac\thelabidx"))^^J
6121 \space\space(markup-locoref^^J
6122 \eledmacmarkuplocorefdepth^^J
6123 :open "\string\hyperlink%
6124 \ifledRcol R\fi%
6125 {\edindexlab\thelabidx}%
6126 {\ifdefempty{\@index@command}%
6127 }%
6128 {\@backslashchar\@index@command}%
6129 {"^^J
6130 :close "})"^^J
6131 :attr "eledmac\thelabidx"^^J
6132 )

```

```

6133     }%
6134     }%
6135 %

```

And now, in any other case.

```

6136 \else%
6137   \gdef\@index@txt{#1}%
6138   \gdef\@edindex@hyperref{}%
6139   \fi%
6140 }%
6141 }
6142 %

```

## 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.

```

6143 \newcommand{\led@set@index@fornote}[1]{%
6144   \ifbool{indtl@innote}%
6145     {\let\index\nindex}%
6146     {}%
6147   \ifbool{indtl@notenumber}%
6148     {%
6149     \renewcommand{\index}[2][\indtl@jobname]{%
6150       \orig@@index[##1]{%
6151         ##2|innotenumber{\csuse{thefootnote#1}}}%
6152       }%
6153     }%
6154   }%
6155   {}%
6156 }%
6157 %

```

`\led@reinit@index@fornote` The `\led@reinit@index@fornote` just reset the default value of `\index`.

```

6158 \newcommand{\led@reinit@index@fornote}{%
6159   \ifbool{indtl@innote}%
6160     {\let\index\orig@@index}%
6161     {}%
6162   \ifbool{indtl@notenumber}%
6163     {\let\index\orig@@index}%
6164     {}%
6165 }%
6166 %

```

## XXVII Glossaries

Here, we define the \gls-Like commands prefixed by ed.

```
\edgls67 \DeclareRobustCommand{\edgls}[3][1,3,usedefault]{%
6168   \doedindexlabel%
6169   \gls[counter=pageline,#1]{#2}[#3]%
6170 }
6171 %
```

```
\edGls72 \DeclareRobustCommand{\edGls}[3][1,3,usedefault]{%
6173   \doedindexlabel%
6174   \Gls[counter=pageline,#1]{#2}[#3]%
6175 }
6176 %
```

```
\edGLS77 \DeclareRobustCommand{\edGLS}[3][1,3,usedefault]{%
6178   \doedindexlabel%
6179   \GLS[counter=pageline,#1]{#2}[#3]%
6180 }
6181 %
```

```
\edglspl82 \DeclareRobustCommand{\edglspl}[3][1,3,usedefault]{%
6183   \doedindexlabel%
6184   \glspl[counter=pageline,#1]{#2}[#3]%
6185 }
6186 %
```

```
\edGlspl87 \DeclareRobustCommand{\edGlspl}[3][1,3,usedefault]{%
6188   \doedindexlabel%
6189   \Glspl[counter=pageline,#1]{#2}[#3]%
6190 }
6191 %
```

```
\edGLSpl92 \DeclareRobustCommand{\edGLSpl}[3][1,3,usedefault]{%
6193   \doedindexlabel%
6194   \GLSpl[counter=pageline,#1]{#2}[#3]%
6195 }
6196 %
```

```
\edglsdisp97 \DeclareRobustCommand{\edglsdisp}[3][1,3,usedefault]{%
6198   \doedindexlabel%
6199   \glsdisp[counter=pageline,#1]{#2}[#3]%
6200 }
6201 %
```

## 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.

```
6202 \def\@hangingsymbol{}
6203 \newcommand*\sethangingsymbol[1]{%
6204   \gdef\@hangingsymbol{#1}%
6205 }%
6206 \newif\ifinstanza
6207 %
```

`\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.

```
6208 \newif\ifinserthangingsymbol
6209 \newcommand{\inserthangingsymbol}{%
6210   \ifinserthangingsymbol%
6211     \ifinstanza%
6212       \@hangingsymbol%
6213     \fi%
6214   \fi%
6215 }
6216 %
```

### 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.

```
6217 \newcommand*\ampersand{\char`\&}
6218
6219 %
```

### 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.

```
6220 \chardef\body=\catcode`\@
6221 \catcode`\@=11
6222 \chardef\next=\catcode`\&
6223 \catcode`\&=\active
6224
6225 %
```

### 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.

```
6226 \newcount\stanza@count
6227 \newlength{\stanzaindentbase}
6228 \setlength{\stanzaindentbase}{20pt}
6229
6230 %
```

`\strip@szacnt` The indentations of stanza lines are non-negative integer multiples of the unit called  
`\setstanzavalues` `\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.

```
6231 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
6232 \newcommand*\setstanzavalues[2]{\def\@tempa{#2,,|}%
6233   \stanza@count\z@
6234   \def\next{\expandafter\strip@szacnt\@tempa
6235     \ifx\@tempb\empty\let\next\relax\else
6236     \expandafter\mathchardef\csname #1@\number\stanza@count
6237     \@endcsname\@tempb\relax
6238     \advance\stanza@count\@ne\fi\next}%
6239   \next}
6240
6241 %
```

`\setstanzaindents` In the original edmac, `\setstanzavalues{sza}{\langle...\rangle}` had to be called to set the in-  
`\setstanzapenalties` dents, and similarly `\setstanzavalues{szp}{\langle...\rangle}` 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).

```
6242 \newcommand*\setstanzaindents[1]{\setstanzavalues{sza}{#1}}
6243 \newcommand*\setstanzapenalties[1]{\setstanzavalues{szp}{#1}}
```



```
6244 %
6245 %
```

**\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.

```
6246 \newcounter{stanzaindentsrepetition}
6247 \newcount\stanza@modulo
6248
6249 \newcommand*{\managestanza@modulo}[0]{%
6250   \advance\stanza@modulo\@ne%
6251   \ifnum\stanza@modulo>\value{stanzaindentsrepetition}%
6252     \stanza@modulo\@ne%
6253   \fi%
6254 }
6255 %
```

**\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.

```
6256 \newcommand{\stanzaindent}[1]{%
6257   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
6258   \ignorespaces%
6259 }%
6260 \WithSuffix\newcommand\stanzaindent*[1]{%
6261   \stanzaindent{#1}%
6262   \global\advance\stanza@modulo-\@ne%
6263   \ifnum\stanza@modulo=0%
6264     \global\stanza@modulo=\value{stanzaindentsrepetition}%
6265   \fi%
6266   \ignorespaces%
6267 }%
6268 %
```

## XXVIII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```
\thestanza69 \newcounter{stanza}
6270 \renewcommand{\thestanza}{%
6271   \textbf{\arabic{stanza}}%
6272 }
6273 %
```

`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```
6274 \newif\ifnumberstanza%
6275 %
```

`\@insertstanzanumber` Now, macro called at the first line of of verse of a stanza.

```
6276 \newcommand{\@insertstanzanumber}[0]{%
6277   \ifnumberstanza%
6278     \ifl@dpairing%
6279       \ifledRcol%
6280         \stanzanumwrapper{\thestanzaR}%
6281       \else%
6282         \stanzanumwrapper{\thestanzaL}%
6283       \fi%
6284     \else%
6285       \stanzanumwrapper{\thestanza}%
6286     \fi%
6287     \setline{1}%
6288   \fi%
6289 }%
6290 %
```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```
6291 \newcommand{\@advancestanzanumber}[0]{%
6292   \ifnumberstanza%
6293     \ifl@dpairing%
6294       \ifledRcol%
6295         \addtocounter{stanzaR}{1}%
6296       \else%
6297         \addtocounter{stanzaL}{1}%
6298       \fi%
6299     \else%
6300       \addtocounter{stanza}{1}%
6301     \fi%
6302   \fi%
6303 }%
6304 %
```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```
6305 \newcommand{\stanzanumwrapper}[1]{%
6306   \flagstanza{#1}%
6307 }%
6308 %
```

## XXVIII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

6309 \newcommand{\printstanza}[0]{%
6310   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
6311     \ifledRcol{%
6312       \thestanzaR%
6313     }else%
6314       \thestanzaL%
6315     \fi%
6316   }{%
6317     \thestanza%
6318   }%
6319 }
6320 %

```

## XXVIII.7 Main work

`\stanza@line` Now we arrive at the main works. `\stanza@line` sets the indentation for the line and 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.

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.

```

6321 \newcommandx{\stanza@line}[1][1]{
6322   \ifnum\value{stanzaindentrepetition}=0
6323     \parindent=\csname sza@number\stanza@count
6324       @\endcsname\stanzaindentbase
6325   \else
6326     \parindent=\csname sza@number\stanza@modulo
6327       @\endcsname\stanzaindentbase
6328     \managestanza@modulo
6329   \fi
6330   \pstart[#1]\stanza@hang\ignorespaces}
6331 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
6332   \hangindent\expandafter
6333   \noexpand\csname sza@0@\endcsname\stanzaindentbase
6334   \hangafter\@ne}
6335 \def\sza@penalty{\count@\csname szp@number\stanza@count @\endcsname
6336   \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
6337   \penalty\fi\count@}
6338 %

```

`\@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 `\&`.

```

6339 \xdef\@startstanza[#1]{%
6340   \noexpand\instanzatrue\expandafter
6341   \begingroup%
6342   \catcode`\noexpand\&\active%
6343   \global\stanza@count\@ne\stanza@modulo\@ne
6344   \noexpand\ifnum\expandafter\noexpand
6345   \csname sza@0\endcsname=\z@\let\noexpand\stanza@hang\relax
6346   \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
6347   \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
6348   \expandafter\noexpand\csname szp@0\endcsname=\z@
6349   \let\noexpand\sza@penalty\relax\noexpand\fi%
6350   \def\noexpand&{%
6351     \noexpand\newverse[] []}%
6352   \def\noexpand\&{\noexpand\@stopstanza}%
6353   \noexpand\@advancestanzanumber%
6354   \noexpand\stanza@line[#1]\noexpand\@insertstanzanumber%
6355   \let\par\relax\ignorespaces%No paragraph in verses
6356 }
6357
6358 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
6359
6360 \newcommandx{\@stopstanza}[1][1,usedefault]{%
6361   \unskip%
6362   \endlock%
6363   \pend[#1]%
6364   \endgroup%
6365   \instanzafalse%
6366 }
6367
6368 \newcommandx*{\newverse}[2][1,2,usedefault]{%
6369   \unskip%
6370   \endlock\pend[#1]\sza@penalty\global%
6371   \advance\stanza@count\@ne\stanza@line[#2]%
6372 }
6373
6374 %

```

**\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`.

```

6375 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
6376   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
6377
6378 %

```

## 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.

```
6379 \catcode`\&=\next
6380 \catcode`\@=\body
6381 \setstanzavalues{szp}{0}
6382
6383 %
```

## 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.

```
6384 \newtoks\@emptytoks
6385
6386 %
```

The rest is from `amsmath`.

`\l@denbody` A token register to contain the body.

```
6387 \newtoks\l@denbody
6388
6389 %
```

`\addtol@denbody` `\addtol@denbody{arg}` adds `arg` to the token register `\l@denbody`.

```
6390 \newcommand{\addtol@denbody}[1]{%
6391 \global\l@denbody\expandafter{\the\l@denbody#1}}
6392
6393 %
```

`\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`.

```

6394 \newcommand{\l@dcollect@body}[1]{%
6395   \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}%
6396   \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currenvir}}%
6397   \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
6398   \begingroup
6399     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
6400     \edef\processl@denvbody{\expandafter\noexpand\csname\@currenvir\
endcsname}%
6401     \processl@denvbody%
6402   }%
6403
6404 %

```

`\l@dpush@begins` When adding a piece of the current environment's contents to `\l@denvbody`, we scan it to check for additional `\begin` tokens, and add a 'b' to the stack for any that we find.

```

6405 \def\l@dpush@begins#1\begin#2{%
6406   \ifx\end#2\else b\expandafter\l@dpush@begins\fi
6407
6408 %

```

`\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.

```

6409 \def\l@dcollect@@body#1\end#2{%
6410   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
6411     \expandafter\@gobble\l@dbegin@stack}%
6412   \ifx\@empty\l@dbegin@stack
6413     \endgroup
6414     \@checkend{#2}%
6415     \addtol@denvbody{#1}%
6416   \else
6417     \addtol@denvbody{#1\end{#2}}%
6418   \fi
6419   \processl@denvbody % A little tricky! Note the grouping
6420 }
6421
6422 %

```

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#{%
  \@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.

```

6423 \newcommand*{\l@dtabnoexpands}{%
6424   \let\rtab=0%
6425   \let\ctab=0%
6426   \let\ltab=0%
6427   \let\rtabtext=0%
6428   \let\ltabtext=0%
6429   \let\ctabtext=0%
6430   \let\edbeforetab=0%
6431   \let\edaftertab=0%
6432   \let\edatleft=0%
6433   \let\edatright=0%
6434   \let\edvertline=0%
6435   \let\edvertdots=0%
6436   \let\edrowfill=0%
6437 }
6438 %
6439 %

```

`\disable@familiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in `edtabularx` and `edarrayx` environments.

```

6440 \newcommand{\disable@familiarnotes}{%
6441   \unless\ifnofamiliar%
6442     \def\do##1{%
6443       \csletcs{footnote@@##1}{footnote##1}%
6444       \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
6445         \protected@csxdef{@thefnmark##1}{\csuse{thefootnote##1}}%
6446         \csuse{@footnotemark##1}%
6447       }%
6448     }%
6449     \dolistloop{\@series}%
6450   \fi%
6451 }%
6452 \newcommand{\restore@familiarnotes}{%
6453   \unless\ifnofamiliar%
6454     \def\do##1{%
6455       \csletcs{footnote##1}{footnote@@##1}%
6456     }%
6457     \dolistloop{\@series}%
6458   \fi%

```

```

6459 }%
6460
6461 %

```

**\disable@sidenotes** The same, for side notes.

**\restore@sidenotes**

```

6462 \newcommand{\disable@sidenotes}{%
6463   \let\@@ledrightnote\ledrightnote%
6464   \let\@@ledleftnote\ledleftnote%
6465   \let\@@ledsidenote\ledsidenote%
6466   \let\ledrightnote@gobble%
6467   \let\ledleftnote@gobble%
6468   \let\ledsidenote@gobble%
6469 }%
6470 \newcommand{\restore@sidenotes}{%
6471   \let\ledrightnote\@@ledrightnote%
6472   \let\ledleftnote\@@ledleftnote%
6473   \let\ledsidenote\@@ledsidenote%
6474 }%
6475 %

```

**\disable@notes** Disable/restore side and familiar notes.

**\restore@notes**

```

6476 \newcommand{\disable@notes}{%
6477   \disable@sidenotes%
6478   \disable@familiarnotes%
6479 }%
6480 \newcommand{\restore@notes}{%
6481   \restore@sidenotes%
6482   \restore@familiarnotes%
6483 }%
6484 %

```

**\EDTEXT** We need to be able to modify the `\edtext` macros and also restore their original definitions.

**\xedtext**

```

6485 \let\EDTEXT=\edtext
6486 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
6487 %

```

**\EDLABEL** We need to be able to modify and restore the `\edlabel` macro.

**\xedlabel**

```

6488 \let\EDLABEL=\edlabel
6489 \newcommand*{\xedlabel}[1]{\EDLABEL{#1}}
6490 %

```

**\EDINDEX** Macros supporting modification and restoration of `\edindex`.

**\xedindex**

**\nulledindex**

```

6491 \let\EDINDEX=\edindex
6492 \newcommand{\xedindex}{\@bsphack%
6493   \@ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
6494 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
6495
6496 %

```

**\@line@num** Macro supporting restoration of \linenum.

```

6497 \let\@line@num=\linenum
6498 %

```

**\l@gobblearg** **\l@gobbleoptarg**[<arg>]{<arg>} replaces these two arguments (first is optional) by \relax.

```

6499 \newcommand*{\l@gobbleoptarg}[2][\relax]%
6500
6501 %

```

**\Relax**<sub>02</sub> \let\Relax=\relax

**\NEXT**<sub>03</sub> \let\NEXT=\next

```

6504
6505 %

```

**\l@dmodforedtext** Modify and restore various macros for when \edtext is used.

**\l@drestoreforedtext**

```

6506 \newcommand{\l@dmodforedtext}{%
6507   \let\edtext\relax
6508   \def\do##1{\global\csletcs{##1footnote}{\l@gobbleoptarg}}%
6509   \dolistloop{\@series}%
6510   \let\edindex\nulledindex
6511   \let\linenum\@gobble}
6512 \newcommand{\l@drestoreforedtext}{%
6513   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
6514   \dolistloop{\@series}%
6515   \let\edindex\xedindex}
6516 %

```

**\l@dnullfills** Nullify and restore some column fillers, etc.

**\l@drestorefills**

```

6517 \newcommand{\l@dnullfills}{%
6518   \def\edlabel##1{%
6519     \def\edrowfill##1##2##3{%
6520     }
6521   \newcommand{\l@drestorefills}{%
6522     \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
6523   }
6524
6525 %

```

`\letsforverteilen` Gathers some lets and other code that is common to the `*verteilen*` macros.

```

6526 \newcommand{\letsforverteilen}{%
6527   \let\edtext\xedtext
6528   \let\edindex\xedindex
6529   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
6530   \dolistloop{\@series}%
6531   \let\linenum\@line@num
6532   \hilfe skip=\l@dcwidth%
6533   \advance\hilfe skip by -\wd\hilfe box
6534   \def\edlabel##1{\xílabel{##1}}
6535 }
6536 %

```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for  
`\enablel@dtabfeet` `\edtext`.

```

6537 \newcommand\disablel@dtabfeet{\l@dmoforedtext}%
6538 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
6539 %

```

### XXIX.2.2 Counters, boxes and lengths

`\l@dampcount` `\l@dampcount` is a counter for the & column dividers and `\l@dcolcount` is a counter for the columns.

```

6540 \newcount\l@dampcount
6541 \l@dampcount=1\relax
6542 \newcount\l@dcolcount
6543 \l@dcolcount=0\relax
6544
6545 %

```

`\hilfe box` Some (temporary) helper items.

```

\hilfe skip
6546 \newbox\hilfe box
\hilfe box
6547 \newskip\hilfe skip
\hilfe count
6548 \newbox\hilfe box
6549 \newcount\hilfe count
6550
6551 %

```

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).

```

6552 \newdimen\dcoli
6553 \newdimen\dcolii
6554 \newdimen\dcoliii
6555 \newdimen\dcoliv
6556 \newdimen\dcolv

```

```

6557 \newdimen\dc colvi
6558 \newdimen\dc colvii
6559 \newdimen\dc colviii
6560 \newdimen\dc colix
6561 \newdimen\dc colx
6562 \newdimen\dc colxi
6563 \newdimen\dc colxii
6564 \newdimen\dc colxiii
6565 \newdimen\dc colxiv
6566 \newdimen\dc colxv
6567 \newdimen\dc colxvi
6568 \newdimen\dc colxvii
6569 \newdimen\dc colxviii
6570 \newdimen\dc colxix
6571 \newdimen\dc colxx
6572 \newdimen\dc colxxi
6573 \newdimen\dc colxxii
6574 \newdimen\dc colxxiii
6575 \newdimen\dc colxxiv
6576 \newdimen\dc colxxv
6577 \newdimen\dc colxxvi
6578 \newdimen\dc colxxvii
6579 \newdimen\dc colxxviii
6580 \newdimen\dc colxxix
6581 \newdimen\dc colxxx
6582 \newdimen\dc colerr % added for error handling
6583
6584 %

```

**\l@dc colwidth** This is a cunning way of storing the columnwidths indexed by the column number \l@dc colcount, like an array. (was \Dimenzuordnung)

```

6585 \newcommand{\l@dc colwidth}{\ifcase \the\l@dc colcount \dc oli %???
6586 \or \dc oli \or \dc olii \or \dc oliii
6587 \or \dc oliv \or \dc olv \or \dc olvi
6588 \or \dc olvii \or \dc olviii \or \dc olvix \or \dc olx
6589 \or \dc olxi \or \dc olxii \or \dc olxiii
6590 \or \dc olxiv \or \dc olxv \or \dc olxvi
6591 \or \dc olxvii \or \dc olxviii \or \dc olxix \or \dc olxx
6592 \or \dc olxxi \or \dc olxxii \or \dc olxxiii
6593 \or \dc olxxiv \or \dc olxxv \or \dc olxxvi
6594 \or \dc olxxvii \or \dc olxxviii \or \dc olxxix \or \dc olxxx
6595 \else \dc olerr \fi}
6596
6597 %

```

**\stepl@dc colcount** This increments the column counter, and issues an error message if it is too large.

```

6598 \newcommand*{\stepl@dc colcount}{\advance\l@dc colcount\@ne
6599 \ifnum\l@dc colcount>30\relax

```

```

6600 \led@err@TooManyColumns
6601 \fi}
6602
6603 %

```

**\l@setmaxcolwidth** Sets the column width to the maximum value seen so far.

```

6604 \newcommand{\l@setmaxcolwidth}{%
6605 \ifdim\l@dcwidth < \wd\hifsbox
6606 \l@dcwidth = \wd\hifsbox
6607 \else \relax \fi}
6608
6609 %

```

**\measurecell** Measure (recursively) the width required for a math cell.

```

6610 \def\measurecell #1{%
6611 \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
6612 \else\l@dcheckcols%
6613 \l@dcwidth=0%
6614 \let\NEXT\measurecell%
6615 \fi%
6616 \else\setbox\hifsbox=\hbox{$\displaystyle{#1}$}%
6617 \step\l@dcwidth%
6618 \l@setmaxcolwidth%
6619 \let\NEXT\measurecell%
6620 \fi\NEXT}
6621
6622 %

```

**\measuretextcell** Measure (recursively) the width required for a text cell.

```

6623 \def\measuretextcell #1{%
6624 \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
6625 \else\l@dcheckcols%
6626 \l@dcwidth=0%
6627 \let\NEXT\measuretextcell%
6628 \fi%
6629 \else\setbox\hifsbox=\hbox{#1}%
6630 \step\l@dcwidth%
6631 \l@setmaxcolwidth%
6632 \let\NEXT\measuretextcell%
6633 \fi\NEXT}
6634
6635 %

```

**\measurerow** Measure (recursively) the width required for a math row.

```

6636 \def\measurerow #1\{%
6637 \ifx #1\let\NEXT\relax%

```

```

6638 \else\measurecell #1\\&\\&%
6639 \let\NEXT\measurerow%
6640 \fi\NEXT}
6641 %

```

**\measurerow** Measure (recursively) the width required for a text row.

```

6642 \def\measurerow #1\{ %
6643 \ifx #1\let\NEXT\relax%
6644 \else\measurecell #1\\&\\&%
6645 \let\NEXT\measurerow%
6646 \fi\NEXT}
6647
6648 %

```

**\edtabcolsep** The length \edtabcolsep controls the distance between columns.

```

6649 \newskip\edtabcolsep
6650 \global\edtabcolsep=10pt
6651
6652 %

```

**\variab**<sub>53</sub> \newcommand{\variab}{\relax}

```

6654
6655 %

```

**\l@dcheckcols** Check that the number of columns is consistent.

```

6656 \newcommand*{\l@dcheckcols}{%
6657 \ifnum\l@dcolcount=1\relax
6658 \else
6659 \ifnum\l@dampcount=1\relax
6660 \else
6661 \ifnum\l@dcolcount=\l@dampcount\relax
6662 \else
6663 \l@d@err@UnequalColumns
6664 \fi
6665 \fi
6666 \l@dampcount=\l@dcolcount
6667 \fi}
6668
6669 %

```

**\edfilldimen** A length.

```

6670 \newdimen\edfilldimen
6671 \edfilldimen=0pt
6672
6673 %

```

`\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`

```
6674 \newcounter{addcolcount}
6675 \renewcommand{\theadcolcount}{\romannumeral \c@addcolcount}
6676 %
```

### XXIX.2.3 Tabular typesetting

`\setmcellright` Typeset (recursively) cells of display math right justified.

```
6677 \def\setmcellright #1{\def\edlabel##1{}%
6678 \let\edindex\nulledindex
6679 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
6680 \let\Next\relax%
6681 \else\l@dcolcount=0%
6682 \let\Next=\setmcellright%
6683 \fi%
6684 \else%
6685 \disablel@dtabfeet%
6686 \step1@dcolcount%
6687 \disable@notes%
6688 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6689 \restore@notes%
6690 \letsforverteilen%
6691 \hskip\hilfsskip$\displaystyle{#1}$%
6692 \hskip\edtabcolsep%
6693 \let\Next=\setmcellright%
6694 \fi\Next}
6695 %
6696 %
```

`\settcclright` Typeset (recursively) cells of text right justified.

```
6697 \def\settcclright #1{\def\edlabel##1{}%
6698 \let\edindex\nulledindex
6699 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
6700 \let\Next\relax%
6701 \else\l@dcolcount=0%
6702 \let\Next=\settcclright%
6703 \fi%
6704 \else%
6705 \disablel@dtabfeet%
6706 \step1@dcolcount%
6707 \disable@notes%
6708 \setbox\hilfsbox=\hbox{#1}%
6709 \restore@notes%
6710 \letsforverteilen%
6711 \hskip\hilfsskip#1%
```



```

6712         \hskip\edtabcolsep%
6713         \let\Next=\settcellright%
6714     \fi\Next}
6715 %

```

**\setmcellleft** Typeset (recursively) cells of display math left justified.

```

6716 \def\setmcellleft #1&{\def\edlabel##1{}}%
6717     \let\edindex\nulledindex
6718     \ifx #1\\ \ifnum\l@dc@colcount=0 \let\Next\relax%
6719         \else\l@dc@colcount=0%
6720             \let\Next=\setmcellleft%
6721         \fi%
6722     \else \disablel@dtabfeet%
6723         \stepl@dc@colcount%
6724         \disable@notes%
6725         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6726         \restore@notes%
6727         \letsforverteilen%
6728         $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
6729         \let\Next=\setmcellleft%
6730     \fi\Next}
6731
6732 %

```

**\settcellleft** Typeset (recursively) cells of text left justified.

```

6733 \def\settcellleft #1&{\def\edlabel##1{}}%
6734     \let\edindex\nulledindex
6735     \ifx #1\\ \ifnum\l@dc@colcount=0 \let\Next\relax%
6736         \else\l@dc@colcount=0%
6737             \let\Next=\settcellleft%
6738         \fi%
6739     \else \disablel@dtabfeet%
6740         \stepl@dc@colcount%
6741         \disable@notes%
6742         \setbox\hilfsbox=\hbox{#1}%
6743         \restore@notes%
6744         \letsforverteilen%
6745         #1\hskip\hilfsskip\hskip\edtabcolsep%
6746         \let\Next=\settcellleft%
6747     \fi\Next}
6748 %

```

**\setmcellcenter** Typeset (recursively) cells of display math centered.

```

6749 \def\setmcellcenter #1&{\def\edlabel##1{}}%
6750     \let\edindex\nulledindex
6751     \ifx #1\\ \ifnum\l@dc@colcount=0\let\Next\relax%
6752         \else\l@dc@colcount=0%

```

```

6753         \let\Next=\setmcellcenter%
6754     \fi%
6755 \else    \disablel@dtabfeet%
6756         \stepl@dcolcount%
6757         \disable@notes%
6758         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6759         \restore@notes%
6760         \letsforverteilen%
6761         \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6762         \hskip\edtabcolsep%
6763         \let\Next=\setmcellcenter%
6764 \fi\Next}
6765
6766 %

```

**\settcclcenter** Typeset (recursively) cells of text centered.

```

6767 \def\settcclcenter #1&{\def\edlabel##1{}%
6768     \let\edindex\nulledindex
6769     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
6770         \else\l@dcolcount=0%
6771             \let\Next=\settcclcenter%
6772         \fi%
6773     \else    \disablel@dtabfeet%
6774             \stepl@dcolcount%
6775             \disable@notes%
6776             \setbox\hilfsbox=\hbox{#1}%
6777             \restore@notes%
6778             \letsforverteilen%
6779             \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6780             \hskip\edtabcolsep%
6781             \let\Next=\settcclcenter%
6782     \fi\Next}
6783
6784 %

```

**\NEXT** \let\NEXT=\relax

```

6785
6786
6787 %

```

**\setmrowright** Typeset (recursively) rows of right justified math.

```

6788 \def\setmrowright #1\\{%
6789     \ifx #1& \let\NEXT\relax
6790     \else \centerline{\setmcellright #1&\\&\\&}
6791     \let\NEXT=\setmrowright
6792 \fi\NEXT}
6793 %

```

**\settroright** Typeset (recursively) rows of right justified text.

```

6794 \def\settroright #1\{\%
6795   \ifx #1& \let\NEXT\relax
6796   \else \centerline{\settcclright #1&\&\&\&}
6797   \let\NEXT=\settroright
6798   \fi\NEXT}
6799
6800 %

```

**\setmrowleft** Typeset (recursively) rows of left justified math.

```

6801 \def\setmrowleft #1\{\%
6802   \ifx #1& \let\NEXT\relax
6803   \else \centerline{\setmcclleft #1&\&\&\&}
6804   \let\NEXT=\setmrowleft
6805   \fi\NEXT}
6806 %

```

**\settrorleft** Typeset (recursively) rows of left justified text.

```

6807 \def\settrorleft #1\{\%
6808   \ifx #1& \let\NEXT\relax
6809   \else \centerline{\settcclleft #1&\&\&\&}
6810   \let\NEXT=\settrorleft
6811   \fi\NEXT}
6812
6813 %

```

**\setmrowcenter** Typeset (recursively) rows of centered math.

```

6814 \def\setmrowcenter #1\{\%
6815   \ifx #1& \let\NEXT\relax%
6816   \else \centerline{\setmcclcenter #1&\&\&\&}
6817   \let\NEXT=\setmrowcenter
6818   \fi\NEXT}
6819 %

```

**\settrorcenter** Typeset (recursively) rows of centered text.

```

6820 \def\settrorcenter #1\{\%
6821   \ifx #1& \let\NEXT\relax
6822   \else \centerline{\settcclcenter #1&\&\&\&}
6823   \let\NEXT=\settrorcenter
6824   \fi\NEXT}
6825
6826 %

```

**\nullsetzen** `\newcommand{\nullsetzen}{%`

```

6828     \step1@dcolcount%
6829     \l@dcolwidth=0pt%
6830     \ifnum\l@dcolcount=30\let\NEXT\relax%
6831         \l@dcolcount=0\relax
6832     \else\let\NEXT\nullsetzen%
6833     \fi\NEXT}
6834
6835 %
```

**\edatleft** `\edatleft[ $\langle math \rangle$ ]{ $\langle symbol \rangle$ }{ $\langle len \rangle$ }. Left  $\langle symbol \rangle$ ,  $2\langle len \rangle$  high with prepended  $\langle math \rangle$  vertically centered.`

```

6836 \newcommand{\edatleft}[3][\@empty]{%
6837     \ifx#1\@empty
6838         \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
6839             depth 0pt \right. $\hss}\vfil}
6840     \else
6841         \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
6842             depth 0pt \right. $\}\vfil}
6843     \fi}
6844 %
```

**\edatright** `\edatright[ $\langle math \rangle$ ]{ $\langle symbol \rangle$ }{ $\langle len \rangle$ }. Right  $\langle symbol \rangle$ ,  $2\langle len \rangle$  high with appended  $\langle math \rangle$  vertically centered.`

```

6845 \newcommand{\edatright}[3][\@empty]{%
6846     \ifx#1\@empty
6847         \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
6848             depth 0pt \right#2 $\hss}\vfil}
6849     \else
6850         \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
6851             depth 0pt \right#2 #1 $\}\vfil}
6852     \fi}
6853
6854 %
```

**\edvertline** `\edvertline{ $\langle len \rangle$ }` vertical line  $\langle len \rangle$  high.

```

6855 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
6856
6857 %
```

**\edvertdots** `\edvertdots{ $\langle len \rangle$ }` vertical dotted line  $\langle len \rangle$  high.

```

6858 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6859     {\cleaders\hbox{$\math\hbox{.}\vbox to 0.5em{ }$\}\vfil}}}
6860
6861 %
```

`\l@dtabaddcols` `\l@dtabaddcols{<startcol>}{<endcol>}` adds the widths of the columns `<startcol>` through `<endcol>` to `\edfilldimen`. It is a  $\TeX$  style reimplementation of the original `\@add@`.

```

6862 \newcommand{\l@dtabaddcols}[2]{%
6863   \l@dcheckstartend{#1}{#2}%
6864   \ifl@dstartendok
6865     \setcounter{addcolcount}{#1}%
6866     \@whilenum \value{addcolcount}<#2\relax \do
6867     {\advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
6868      \advance\edfilldimen by \edtabcolsep
6869      \stepcounter{addcolcount}}%
6870     \advance\edfilldimen by \the\csname dcol\theaddcolcount\endcsname
6871     \fi
6872   }
6873   %
6874   %

```

`\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.

```

6875 \newif\ifl@dstartendok
6876 \newcommand{\l@dcheckstartend}[2]{%
6877   \l@dstartendoktrue
6878   \ifnum #1<\@ne
6879     \l@dstartendokfalse
6880     \led@err@LowStartColumn
6881   \fi
6882   \ifnum #2>30\relax
6883     \l@dstartendokfalse
6884     \led@err@HighEndColumn
6885   \fi
6886   \ifnum #1>#2\relax
6887     \l@dstartendokfalse
6888     \led@err@ReverseColumns
6889   \fi
6890 }
6891 %
6892 %

```

`\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.

```

6893 \newcommand*\edrowfill}[3]{%
6894   \l@dtabaddcols{#1}{#2}%
6895   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
6896   \let\@edrowfill@=\edrowfill
6897   \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}

```

```
6898
6899 %
```

**\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`.

```
6900 \newcommand{\leftltab}[1]{%
6901   \hb@xt@{z@{\vbox{\edtabindent%
6902     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6903
6904 %
```

**\leftrtab**      `\leftrtab{<text>}{<math>}` for `\edbeforetab` in `\rtab`.

```
6905 \newcommand{\leftrtab}[2]{%
6906   #2\hb@xt@{z@{\vbox{\edtabindent%
6907     \advance\Hilfsskip by\dcoli%
6908     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6909
6910 %
```

**\leftctab**      `\leftctab{<text>}{<math>}` for `\edbeforetab` in `\ctab`.

```
6911 \newcommand{\leftctab}[2]{%
6912   \hb@xt@{z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6913     \advance\Hilfsskip by 0.5\dcoli%
6914     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6915     \disablel@dtabfeet$\displaystyle{#2}$}%
6916     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6917     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
6918   #2}
6919
6920 %
```

**\rightctab**      `\rightctab{<math>}{<text>}` for `\edaftertab` in `\ctab`.

```
6921 \newcommand{\rightctab}[2]{%
6922   \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6923   \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6924   #1\hb@xt@{z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6925     \advance\Hilfsskip by 0.5\l@dcolwidth%
6926     \advance\Hilfsskip by -\wd\hilfsbox%
6927     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6928     \disablel@dtabfeet$\displaystyle{#1}$}%
6929     \advance\Hilfsskip by -0.5\wd\hilfsbox%
```

```

6930 \advance\Hilfsskip by \edtabcolsep%
6931 \moveright\Hilfsskip\hbox{ #2}\hss}%
6932 }
6933
6934 %

```

**\rightltab** `\rightltab{<math><math>}{<text>}` for `\edaftertab` in `\ltab`.

```

6935 \newcommand{\rightltab}[2]{%
6936 \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6937 \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6938 #1\hb@xt@{\z@\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6939 \advance\Hilfsskip by\l@dcolwidth%
6940 \advance\Hilfsskip by-\wd\hilfsbox%
6941 \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6942 \disablel@dtabfeet$\displaystyle{#1}$}%
6943 \advance\Hilfsskip by-\wd\hilfsbox%
6944 \advance\Hilfsskip by\edtabcolsep%
6945 \moveright\Hilfsskip\hbox{ #2}\hss}%
6946 }
6947
6948 %

```

**\rightrtab** `\rightrtab{<math><math>}{<text>}` for `\edaftertab` in `\rtab`.

```

6949 \newcommand{\rightrtab}[2]{%
6950 \setbox\hilfsbox=\hbox{\def\edlabel##1{%
6951 \disablel@dtabfeet#2}%
6952 #1\hb@xt@{\z@\vbox{\edtabindent%
6953 \advance\Hilfsskip by-\wd\hilfsbox%
6954 \advance\Hilfsskip by\edtabcolsep%
6955 \moveright\Hilfsskip\hbox{ #2}\hss}%
6956 }
6957
6958 %

```

**\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.

```

6959 \newcommand{\rtab}[1]{%
6960 \l@dnnullfills
6961 \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
6962 \def\edaftertab##1##2{\righttab{##1}{##2}}%
6963 \measurebody{#1}%
6964 \l@drestorefills
6965 \variab
6966 \setmrowright #1\&\&%
6967 \enablel@dtabfeet}
6968
6969 %

```

`\measurebody` `\measurebody{⟨body⟩}` measures the array `⟨body⟩`.

```

6970 \newcommand{\measurebody}[1]{%
6971   \disablel@dtabfeet%
6972   \l@dcolcount=0%
6973   \nullsetzen%
6974   \l@dcolcount=0
6975   \measuremrow #1\\&\\%
6976   \global\l@dampcount=1}
6977
6978 %

```

`\rtabtext` `\rtabtext{⟨body⟩}` typesets `⟨body⟩` as a tabular with the entries right justified.

```

6979 \newcommand{\rtabtext}[1]{%
6980   \l@dnullfills
6981   \measuretbody{#1}%
6982   \l@drestorefills
6983   \variab
6984   \setthrowright #1\\&\\%
6985   \enablel@dtabfeet}
6986
6987 %

```

`\measuretbody` `\measuretbody{⟨body⟩}` measures the tabular `⟨body⟩`.

```

6988 \newcommand{\measuretbody}[1]{%
6989   \disable@notes%
6990   \disablel@dtabfeet%
6991   \l@dcolcount=0%
6992   \nullsetzen%
6993   \l@dcolcount=0
6994   \measuretrow #1\\&\\%
6995   \restore@notes%
6996   \global\l@dampcount=1}
6997
6998 %

```

`\ltab` Array with entries left justified.

```

\edbeforetab
\edaftertab
6999 \newcommand{\ltab}[1]{%
7000   \l@dnullfills
7001   \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
7002   \def\edaftertab##1##2{\rightltab{##1}{##2}}%
7003   \measuretbody{#1}%
7004   \l@drestorefills
7005   \variab
7006   \setmrowleft #1\\&\\%
7007   \enablel@dtabfeet}
7008
7009 %

```



`\ltabtext` Tabular with entries left justified.

```

7010 \newcommand{\ltabtext}[1]{%
7011   \l@dnnullfills
7012   \measuretbody{#1}%
7013   \l@drestorefills
7014   \variab
7015   \settrorleft #1\&\%
7016   \enablel@dtabfeet}
7017
7018 %

```

`\ctab` Array with centered entries.

```

\edbeforetab
\edaftertab
7019 \newcommand{\ctab}[1]{%
7020   \l@dnnullfills
7021   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
7022   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
7023   \measuretbody{#1}%
7024   \l@drestorefills
7025   \variab
7026   \setmrowcenter #1\&\%
7027   \enablel@dtabfeet}
7028
7029 %

```

`\ctabtext` Tabular with entries centered.

```

7030 \newcommand{\ctabtext}[1]{%
7031   \l@dnnullfills
7032   \measuretbody{#1}%
7033   \l@drestorefills
7034   \variab
7035   \settrorcenter #1\&\%
7036   \enablel@dtabfeet}
7037
7038 %

```

```

\spreadtext
7039 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
7040   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
7041 %

```

```

\spreadmath
7042 \newcommand{\spreadmath}[1]{%
7043   \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
7044
7045 %

```

`\HILFSskip` More helpers.

`\Hilfsskip`

```

7046 \newskip\HILFSskip
7047 \newskip\Hilfsskip
7048
7049 %

```

```

\EDTABINDENT \newcommand{\EDTABINDENT}{%
7050
7051   \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
7052   \else\step\l@dcolcount%
7053     \advance\Hilfsskip by\l@dcolwidth%
7054     \ifdim\l@dcolwidth=0pt\advance\hilfscout\@ne
7055     \else\advance\Hilfsskip by \the\hilfscout\edtabcolsep%
7056     \hilfscout=1\fi%
7057     \let\NEXT=\EDTABINDENT%
7058   \fi\NEXT}%
7059 %

```

**\edtabindent** (was \tabindent)

```

7060 \newcommand{\edtabindent}{%
7061   \l@dcolcount=0\relax
7062   \Hilfsskip=0pt%
7063   \hilfscout=1\relax
7064   \EDTABINDENT%
7065   \hilfsskip=\hsize%
7066   \advance\hilfsskip -\Hilfsskip%
7067   \Hilfsskip=0.5\hilfsskip%
7068 }%
7069
7070 %

```

**\EDTAB** (was \TAB)

```

7071 \def\EDTAB #1|#2|{%
7072   \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
7073   \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
7074   \advance\tabelskip -\wd\tabhilfbox%
7075   \advance\tabelskip -\wd\tabHilfbox%
7076   \unhbox\tabhilfbox\hskip\tabelskip%
7077   \unhbox\tabHilfbox}%
7078
7079 %

```

**\EDTABtext** (was \TABtext)

```

7080 \def\EDTABtext #1|#2|{%
7081   \setbox\tabhilfbox=\hbox{#1}%
7082   \setbox\tabHilfbox=\hbox{#2}%
7083   \advance\tabelskip -\wd\tabhilfbox%
7084   \advance\tabelskip -\wd\tabHilfbox%

```

```

7085 \unhbox\tabhilfbox\hskip\tabelskip%
7086 \unhbox\tabHilfbox}%
7087 %

```

`\tabhilfbox` Further helpers.

```

7088 \newbox\tabhilfbbox
7089 \newbox\tabHilfbbox
7090
7091 %

```

## XXIX.2.4 Environments

edarrayl edarrayc edarrayr The ‘environment’ forms for \ltab, \ctab and \rtab.

```

7092 \newenvironment{edarrayl}{\l@dcollect@body\ltab}{
7093 \newenvironment{edarrayc}{\l@dcollect@body\ctab}{
7094 \newenvironment{edarrayr}{\l@dcollect@body\rtab}{
7095
7096 %

```

edtabularl edtabularc edtabularr The ‘environment’ forms for `\ltabtext`, `\ctabtext` and `\rtabtext`.

```

7097 \newenvironment{edtabularl}{\l@dcollect@body\ltabtext}{\}
7098 \newenvironment{edtabularc}{\l@dcollect@body\ctabtext}{\}
7099 \newenvironment{edtabularr}{\l@dcollect@body\rtabtext}{\}
7100
7101 %

```

### 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
\endquotation
\newcommand{\initnumbering@quote}{
\ifnoquotation@else
\renewcommand{\quotation}{\par\leavevmode%
\parindent=1.5em%
\skipnumbering%
\ifautopar%
\vskip-\parskip%
\else%
\vskip\topsep%
\fi%
\global\leftskip=\leftmargin%

```

```

7113             \global\rightskip=\leftmargin%
7114         }
7115     \renewcommand{\endquotation}{\par%
7116         \global\leftskip=0pt%
7117         \global\rightskip=0pt%
7118         \leavevmode%
7119         \skipnumbering%
7120         \ifautopar%
7121             \vskip-\parskip%
7122         \else%
7123             \vskip\topsep%
7124         \fi%
7125     }
7126     \renewcommand{\quote}{\par\leavevmode%
7127         \parindent=0pt%
7128         \skipnumbering%
7129         \ifautopar%
7130             \vskip-\parskip%
7131         \else%
7132             \vskip\topsep%
7133         \fi%
7134         \global\leftskip=\leftmargin%
7135         \global\rightskip=\leftmargin%
7136     }
7137     \renewcommand{\endquote}{\par%
7138         \global\leftskip=0pt%
7139         \global\rightskip=0pt%
7140         \leavevmode%
7141         \skipnumbering%
7142         \ifautopar%
7143             \vskip-\parskip%
7144         \else%
7145             \vskip\topsep%
7146         \fi%
7147     }
7148     \fi
7149 }
7150 %

```

## XXXI Section's title commands

### XXXI.1 Commands to disable some feature

**\ledsectnotoc** The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```

7151 \newcommand{\ledsectnotoc}{\let\addcontentsline\@gobblethree}
7152 %

```

`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```

7153 \newcommand{\ledsectnomark}{%
7154   \let\chaptermark@gobble%
7155   \let\sectionmark@gobble%
7156   \let\subsectionmark@gobble%
7157 }
7158 %

```

## 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 `ℒTEX` 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 `ℒTEX` 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).

```

7159 \catcode`\#=12
7160 \notbool{@noeled@sec}{%
7161 %

```

`\beforeeledchapter` For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by `eledmac`. Users have to use `\beforeeledchapter`.

```

7162 \ifl@dmemoir
7163   \newcommand\beforeeledchapter{%
7164     \clearforchapter%
7165   }
7166 \else
7167   \newcommand\beforeeledchapter{%
7168     \if@openright%
7169       \cleardoublepage%
7170     \else%
7171       \clearpage%
7172     \fi%
7173   }
7174 \fi
7175 %

```

#### 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.

```

7176 \newif\if@eled@sectioning
7177 %

```

`\print@leftmargin@eledsection` `\print@rightmargin@eledsection` `\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.

```

7178 \def\print@rightmargin@eledsection{%
7179   \if@eled@sectioning%
7180     \begingroup%
7181     \if@RTL%
7182       \let\llap\rlap%
7183       \let\leftlinenum\rightlinenum%
7184       \let\leftlinenumR\rightlinenumR%
7185       \let\l@drd@ta\l@dld@ta%
7186       \let\l@drsn@te\l@dlsn@te%
7187     \fi%
7188     \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
7189     \endgroup%
7190   \fi%
7191 }%
7192
7193 \def\print@leftmargin@eledsection{%
7194   \if@eled@sectioning%
7195     \leavevmode%
7196     \begingroup%

```

```

7197 \if@RTL%
7198     \let\rlap\llap%
7199     \let\rightlinenum\leftlinenum%
7200     \let\rightlinenumR\leftlinenumR%
7201     \let\l@dld@ta\l@dld@ta%
7202     \let\l@dlsn@te\l@dlsn@te%
7203 \fi%
7204 \l@dld@ta\csuse{LR}{\l@dlsn@te}%
7205 \endgroup%
7206 \fi%
7207 }%
7208
7209 %

```

### XXXI.5 Patching standard commands

`\chapter` We have to patch  $\LaTeX$ , book and memoir sectioning commands in order to:

- `\M@sect` • Disable `\edtext` inside.
- `\@mem@old@ssect` • Disable page breaking (for `\chapter`).
- `\@makechapterhead` • Add line numbers and sidenotes.
- `\@makechapterhead`
- `\@makeschapterhead`

`\@sect` Unfortunately, Maïeul Rouquette was not able to try if memoir is loaded. That is why  
`\@ssect` `eledmac` tries to define for both standard class and memoir class.

```

7210 \AtBeginDocument{%
7211 \patchcmd{\chapter}{\clearforchapter}{%
7212 \if@eled@sectioning\else%
7213 \ifl@dprintingpages\else%
7214 \clearforchapter%
7215 \fi%
7216 \fi%
7217 }
7218 {}
7219 {}
7220
7221
7222 \pretocmd{\M@sect}
7223 {\let\old@edtext=\edtext%
7224 \let\edtext=\dummy@edtext@showlemma%
7225 }
7226 {}
7227 {}
7228
7229 \apptocmd{\M@sect}
7230 {\let\edtext=\old@edtext}
7231 {}
7232 {}

```

```

7233 \patchcmd{\M@sect}
7234 { #9}
7235 { #9%
7236 \print@rightmargin@eledsection%
7237 }
7238 {}
7239 {}
7240 {}
7241
7242 \patchcmd{\M@sect}
7243 {\hskip #3\relax}
7244 {\hskip #3\relax%
7245 \print@leftmargin@eledsection%
7246 }
7247 {}
7248 {}
7249
7250 \patchcmd{\@mem@old@ssect}
7251 {#5}
7252 {#5%
7253 \print@leftmargin@eledsection%
7254 }
7255 {}
7256 {}
7257
7258 \patchcmd{\@mem@old@ssect}
7259 {\hskip #1}
7260 {\hskip #1%
7261 \print@rightmargin@eledsection%
7262 }
7263 {}
7264 {}
7265
7266 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
7267 \if@eled@sectioning\else%
7268 \ifl@dprintingpages\else%
7269 \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
7270 \fi%
7271 \fi%
7272 }%
7273 {}%
7274 {}%
7275
7276 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\
fi}{%
7277 \if@eled@sectioning\else%
7278 \ifl@dprintingpages\else%
7279 \if@openright\cleardoublepage\else\clearpage\fi%No clearpage inside a

```



```

\Pages: will keep critical notes from printing on the title page. Here for
scrbook.
    \fi%
7280 \fi%
7281 \fi%
7282 }
7283 {}
7284 {}
7285
7286 \patchcmd{\@makechapterhead}
7287 {#1}
7288 {\print@leftmargin@eledsection%
7289   #1%
7290 \print@rightmargin@eledsection%
7291 }
7292 {}
7293 {}
7294
7295 \patchcmd{\@makechapterhead}% For BIDI
7296 {\if@RTL\raggedleft\else\raggedright\fi}%
7297 {\if@eled@sectioning\else%
7298   \if@RTL\raggedleft\else\raggedright\fi%
7299 \fi%
7300 }%
7301 {}%
7302 {}%
7303
7304 \patchcmd{\@makeschapterhead}
7305 {#1}
7306 {\print@leftmargin@eledsection%
7307   #1%
7308 \print@rightmargin@eledsection%
7309 }
7310 {}
7311 {}
7312
7313 \pretocmd{\@sect}
7314 {\let\old@edtext=\edtext
7315 \let\edtext=\dummy@edtext@showlemma%
7316 }
7317 {}
7318 {}
7319
7320 \apptocmd{\@sect}
7321 {\let\edtext=\old@edtext}
7322 {}
7323 {}
7324
7325 \pretocmd{\@ssect}
7326 {\let\old@edtext=\edtext%
7327 \let\edtext=\dummy@edtext@showlemma%

```

```

7328 }
7329 {}
7330 {}
7331
7332 \apptocmd{\@ssect}
7333 {\let\edtext=\old@edtext}
7334 {}
7335 {}
7336
7337 %

```

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.

```

7338 \@ifpackageloaded{nameref}{
7339
7340   \patchcmd{\NR@sect}
7341     {#8}
7342     {#8%
7343       \print@rightmargin@eledsection%
7344     }
7345     {}
7346     {}
7347
7348   \patchcmd{\NR@sect}
7349     {\hskip #3\relax}
7350     {\hskip #3\relax%
7351       \print@leftmargin@eledsection%
7352     }
7353     {}
7354     {}
7355
7356   \patchcmd{\NR@ssect}
7357     {#5}
7358     {#5%
7359       \print@rightmargin@eledsection%
7360     }
7361     {}
7362     {}
7363
7364   \patchcmd{\NR@ssect}
7365     {\hskip #1}
7366     {\hskip #1%
7367       \print@leftmargin@eledsection%
7368     }
7369     {}
7370     {}
7371   }%
7372   {
7373     \patchcmd{\@sect}
7374       {#8}

```

```

7375     {#8%
7376     \print@rightmargin@eledsection%
7377     }
7378     {}
7379     {}
7380
7381     \patchcmd{\@sect}
7382     {\hskip #3\relax}
7383     {\hskip #3\relax%
7384     \print@leftmargin@eledsection%
7385     }
7386     {}
7387     {}
7388
7389     \patchcmd{\@ssect}
7390     {#5}
7391     {#5%
7392     \print@rightmargin@eledsection%
7393     }
7394     {}
7395     {}
7396
7397     \patchcmd{\@ssect}
7398     {\hskip #1}
7399     {\hskip #1%
7400     \print@leftmargin@eledsection%
7401     }
7402     {}
7403     {}
7404     }%
7405 }
7406 %

```

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.

```

7407 {}}%
7408 \protect\catcode`\#=6 %Space NEEDS by \catcode
7409 \notbool{@noeled@sec}{%
7410 %

```

## 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.

```

7411 \newwrite\eled@sectioning@out
7412 %

```

`\eledchapter` And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

`\eledsection`

`\eledsubsection`

```

7413 \newcommand{\eledchapter}[2] [] {%
7414   #2%
7415   \ifledRcol%
7416     \immediate\write\eled@sectioningR@out{%
7417       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
7418     }%
7419   \else%
7420     \immediate\write\eled@sectioning@out{%
7421       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
7422     }%
7423   \fi%
7424 }
7425
7426 \newcommand{\eledsection}[2] [] {%
7427   #2%
7428   \ifledRcol%
7429     \immediate\write\eled@sectioningR@out{%
7430       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
7431     }%
7432   \else%
7433     \immediate\write\eled@sectioning@out{%
7434       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
7435     }%
7436   \fi%
7437 }
7438
7439 \newcommand{\eledsubsection}[2] [] {%
7440   #2%
7441   \ifledRcol%
7442     \immediate\write\eled@sectioningR@out{%
7443       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{R}
7444     }%
7445   \else%
7446     \immediate\write\eled@sectioning@out{%
7447       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{L}
7448     }%
7449   \fi%
7450 }
7451 \newcommand{\eledsubsubsection}[2] [] {%
7452   #2%
7453   \ifledRcol%
7454     \immediate\write\eled@sectioningR@out{%
7455       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
7456     }%
7457   \else%

```

```

7458 \immediate\write\eled@sectioning@out{%
7459 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
7460 }{}{}
7461 }%
7462 \fi%
7463 }
7464
7465 \WithSuffix\newcommand\eledchapter*[2][]{%
7466 #2%
7467 \ifledRcol%
7468 \immediate\write\eled@sectioningR@out{%
7469 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
7470 }%
7471 \else%
7472 \immediate\write\eled@sectioning@out{%
7473 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dumpstartsL}{*}{L}
7474 }%
7475 \fi%
7476 }
7477
7478 \WithSuffix\newcommand\eledsection*[2][]{%
7479 #2%
7480 \ifledRcol%
7481 \immediate\write\eled@sectioningR@out{%
7482 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
7483 }%
7484 \else%
7485 \immediate\write\eled@sectioning@out{%
7486 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dumpstartsL}{*}{L}
7487 }%
7488 \fi%
7489 }
7490
7491 \WithSuffix\newcommand\eledsubsection*[2][]{%
7492 #2%
7493 \ifledRcol%
7494 \immediate\write\eled@sectioningR@out{%
7495 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
7496 }%
7497 \else%
7498 \immediate\write\eled@sectioning@out{%
7499 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL}
7500 }{*}{L}
7501 }%
7502 \fi%
7503 }
7504 \WithSuffix\newcommand\eledsubsubsection*[2][]{%

```

```

7505 #2%
7506 \ifledRcol%
7507 \immediate\write\eled@sectioningR@out{%
7508 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
7509 }{*}{R}
7509 }%
7510 \else%
7511 \immediate\write\eled@sectioning@out{%
7512 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
7513 }{*}{L}
7513 }%
7514 \fi%
7515 }
7516 %

```

### XXXI.7 Macros written in the auxiliary file

`\eled@chapter`  
`\eled@section`  
`\eled@subsection`  
`\eled@subsubsection`

The sectioning macros, called in the auxiliary file. They have five arguments:

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.

```

7517 \def\eled@chapter#1#2#3#4#5{%
7518 \ifstrempy{#4}%
7519 {%
7520 \ifstrempy{#1}%
7521 {%
7522 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%
7523 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%
7524 }%Need for \pairs, because of using parbox.
7525 {%
7526 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter[#1]{#2}}%
7527 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%Need for \pairs, because of using parbox.
7528 }%
7529 }%
7530 {%
7531 \ifstrempy{#1}%
7532 {%\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%

```

```

7533     {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
7534     }%
7535     \listcsadd{eled@sections#5@@}{#3}%
7536   }
7537   \def\eled@section#1#2#3#4#5{%
7538     \ifstrempy{#4}%
7539     {\ifstrempy{#1}%
7540       {%
7541         \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
7542         \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#2}}}%Need for \pairs, because of using parbox.
7543       }%
7544       {%
7545         \global\csdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
7546         \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
sectionmark{#1}}}%Need for \pairs, because of using parbox.
7547       }%
7548     }%
7549     {\ifstrempy{#1}%
7550       {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
7551       {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
7552     }
7553     \listcsadd{eled@sections#5@@}{#3}%
7554   }
7555   \def\eled@subsection#1#2#3#4#5{%
7556     \ifstrempy{#4}%
7557     {\ifstrempy{#1}%
7558       {%
7559         \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
7560         \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)
7561       }%
7562       {%
7563         \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
7564         \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)
7565       }%
7566     }%
7567     {\ifstrempy{#1}%
7568       {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
7569       {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
7570     }
7571     \listcsadd{eled@sections#5@@}{#3}%
7572   }
7573   \def\eled@subsubsection#1#2#3#4#5{%

```

```

7574 \ifstrempy{#4}%
7575   {\ifstrempy{#1}%
7576     {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
7577     {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#1}{#2}}}%
7578   }%
7579   {\ifstrempy{#1}%
7580     {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
7581     {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
7582   }
7583   \listcsgadd{eled@sections#5@@}{#3}%
7584   }
7585
7586 %

```

End of the conditional test about noeledsec option.

```

7587 }{}
7588 %

```

## 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.

```

7589 \def\normal@page@break{}
7590 %

```

**\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.

```

7591 \def\l@prev@pb{}
7592 \def\l@prev@nopb{}
7593 %

```

**\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.



```

7594 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
7595 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
7596 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
7597 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
7598 %

```

**\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.

```

7599 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
7600 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
7601 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
7602 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
7603 %

```

**\ledpbsetting** The \ledpbsetting macro only changes the value of \led@pb@macro, for which the default value is before.

**\led@pb@setting**

```

7604 \def\led@pb@setting{before}
7605 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
7606 %

```

**\led@check@pb** The \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.

**\led@check@nopb**

```

7607 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
7608 \newcommand{\led@check@nopb}{%
7609   \IfStrEq{\led@pb@setting}{before}{%
7610     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
7611       {\numdef{\abs@prevline}{\the\absline@num-1}}%
7612       \xifinlist{\abs@prevline}{\normal@page@break}%
7613       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7614     }%
7615   }%
7616 }%
7617 }%
7618 \IfStrEq{\led@pb@setting}{after}{%
7619   \xifinlist{\the\absline@num}{\l@prev@nopb}{%
7620     \xifinlist{\the\absline@num}{\normal@page@break}%
7621     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7622   }%
7623 }%
7624 }%
7625 }%
7626 }%
7627 }
7628 %

```

### 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.

```

7629 \newcommand{\check@pb@in@verse}{%
7630   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
enabling page breaks in verse control, while on a hanging verse.
7631   \ifnum\page@num=\last@page@num\else%If we have change page
7632   \IfStrEq{\led@pb@setting}{before}{%
7633     \numgdef{\abs@line@verse}{\the\absline@num-1}%
7634     \ledpbnum{\abs@line@verse}%
7635   }{}%
7636   \IfStrEq{\led@pb@setting}{after}{%
7637     \numgdef{\abs@line@verse}{\the\absline@num-1}%
7638     \lednopbnum{\abs@line@verse}%
7639   }{}%
7640   \fi%
7641   \fi\fi\fi%
7642 }
7643 %

```

### XXXIV Compatibility with eledmac

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

```

7644 \ifeledmaccompat%
7645
7646   \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
7647   \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
7648   \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
7649   \newcommand{\footthreecolX}[1]{\XarrangementX[#1]{threecol}}%
7650
7651   \unless\ifnocritical@
7652     \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
7653     \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
7654     \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
7655     \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%

```

```

7656 \let\hsizetwocol\Xhsizetwocol
7657 \let\hsizethreecol\Xhsizethreecol
7658 \let\bhookXnote\Xbhooknote
7659 \let\boxsymlinenum\Xboxsymlinenum
7660 \let\symlinenum\Xsymlinenum
7661 \let\beforenumberinfootnote\Xbeforenumber
7662 \let\afternumberinfootnote\Xafternumber
7663 \let\beforeXsymlinenum\XbeforeXsymlinenum
7664 \let\afterXsymlinenum\XafterXsymlinenum
7665 \let\inplaceofnumber\Xinplaceofnumber
7666 \let\Xlemmaseparator\lemmaseparator
7667 \let\afterlemmaseparator\Xafterlemmaseparator
7668 \let\beforelemmaseparator\Xbeforelemmaseparator
7669 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
7670 \let\txbeforeXnotes\Xtxbeforenotes
7671 \let\afterXrule\Xafterrule
7672 \let\numberonlyfirstinline\Xnumberonlyfirstinline
7673 \let\numberonlyfirstintwoline\Xnumberonlyfirstintwoline
7674 \let\nonumberinfootnote\Xnonumberinfootnote
7675 \let\pstartinfootnote\Xpstart
7676 \let\pstartinfootnoteeverytime\Xpstarteverytime
7677 \let\onlyXpstart\Xonlypstart
7678 \let\Xnonumberinfootnote\Xnonumber
7679 \let\Xnonbreakableafternumber\Xnonbreakableafternumber
7680 \let\XmaxhXnotes\Xmaxhnotes
7681 \let\beforeXnotes\Xbeforenotes
7682 \let\boxlinenum\Xboxlinenum
7683 \let\boxlinenumalign\Xboxlinenumalign
7684 \let\boxstartlinenum\Xboxstartlinenum
7685 \let\boxendlinenum\Xboxendlinenum
7686 \let\twoline\Xtwoline
7687 \let\morethantwoline\Xmorethantwoline
7688 \let\twolinebutnotmore\Xtwolinebutnotmore
7689 \let\twolineonlyinsamepage\Xtwolineonlyinsamepage
7690 \fi
7691
7692 \unless\ifnofamiliar@
7693 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
7694 \fi
7695 \newcommandx{\parafootsep}[2][1,usedefault]{%
7696 \Xparafootsep[#1]{#2}%
7697 \parafootsepX[#1]{#2}
7698 }%
7699
7700 \newcommandx{\afternote}[2][1,usedefault]{%
7701 \Xafternote[#1]{#2}%
7702 \afternoteX[#1]{#2}%
7703 }%
7704
7705 \unless\ifnoend@

```

```

7706 \let\XendXtwolines\Xendtwolines
7707 \let\XendXmorethantwolines\Xendmorethantwolines
7708 \let\hookXendnote\Xendhooknote
7709 \let\boxXendlinenum\Xendboxlinenum%
7710 \let\boxXendlinenumalign\Xendboxlinenumalign%
7711 \let\boxXendstartlinenum\Xendboxstartlinenum%
7712 \let\boxXendendlinenum\Xendboxendlinenum%
7713 \let\XendXlemmaseparator\Xendlemmaseparator
7714 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
7715 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
7716 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
7717 \fi
7718
7719 \AtBeginDocument{%
7720 \ifdef\lineref{}\let\lineref\edlineref}%
7721 }%
7722
7723
7724 \fi%
7725 %

```

</code>

## Appendix A Some things to do when changing version

### 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>34</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:

<pre>I saw my friend \critext{Smith} \Afootnote{Jones C, D.}/ on Tuesday.</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D.</pre>
---	---

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:

<pre>\critext{I saw my friend \critext{Smith}{\Afootnote{Jones C, D.}/ on Tuesday.} \Bfootnote{The date was July 16, 1954.} /</pre>	<pre>1 I saw my friend 2 Smith on Tuesday. 2 Smith] Jones C, D. 1-2 I saw my friend Smith on Tuesday.] The date was July 16, 1954.</pre>
---	--

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:

<sup>34</sup>A name like `\text` is likely to be defined by other  $\TeX$  packages (it certainly is by the AMS packages) and it seems sensible to try and avoid clashes with other definitions.

```
\catcode`\<=\active
\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. 29) 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>35</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>35</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 introduce 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 `\printnnum`. This spurious space has been deleted. However, if you want to keep the previous spurious space, you may load the package with the `oldprintnnumspace` 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 18.1.1 p. 59 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 `ledsecnolinenumber` option has been removed, because it was related to deprecated commands.

The `oldprintnpnumspace` 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 <code>\renewcommand</code></i>	<i>Replaced with</i>
<code>\@led@extranofeet</code>	<code>\newseries</code>
<code>\apprefprefixmore</code>	<code>\setapprefprefixmore</code>
<code>\apprefprefixsingle</code>	<code>\setapprefprefixsingle</code>
<code>\endstanzaextra</code>	Optional argument of <code>\&amp;</code>
<code>\hangingsymbol</code>	<code>\sethangingsymbol</code>
<code>\ledfootinsdim</code>	<code>\Xmaxhnotes</code> and <code>\maxhnotesX</code>
<code>\parafootftmsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\notenumfont</code>	<code>\Xnotenumfont</code> , <code>\Xendnotenumfont</code> and <code>\notenumfontX</code>
<code>\notefontsetup</code>	<code>\Xnotefontsize</code> , <code>\Xendnotefontsize</code> and <code>\notefontsizeX</code>
<code>\sidenotesep</code>	<code>\setsidenotsep</code>
<code>\startstanzahook</code>	Optional argument of <code>\stanza</code>
<code>\symplinenum</code>	<code>\Xsymplinenum</code>

### 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>
<code>\footparagraph</code>	<code>\Xarrangement</code>
<code>\footnormal</code>	<code>\Xarrangement</code>
<code>\foottwocol</code>	<code>\Xarrangement</code>
<code>\footthreecol</code>	<code>\Xarrangement</code>
<code>\footparagraphX</code>	<code>\arrangementX</code>
<code>\footnormalX</code>	<code>\arrangementX</code>
<code>\foottwocolX</code>	<code>\arrangementX</code>
<code>\footthreecolX</code>	<code>\arrangementX</code>
<code>\afterlemmaseparator</code>	<code>\Xafterlemmaseparator</code>
<code>\afternote</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\afternumberinfootnote</code>	<code>\Xafternumber</code>
<code>\afterXrule</code>	<code>\Xafterrule</code>
<code>\afterXsymplinenum</code>	<code>\Xaftersymplinenum</code>
<code>\beforelemmaseparator</code>	<code>\Xbeforelemmaseparator</code>
<code>\beforenumberinfootnote</code>	<code>\Xbeforenumber</code>
<code>\beforeXnotes</code>	<code>\Xbeforenotes</code>
<code>\beforeXsymplinenum</code>	<code>\Xbeforesymplinenum</code>

<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 `\Xlinerrangeseparator` or `/` and `\Xendlinerrangeseparator` (6.2.3 p. 31).

### Appendix A.14 Migration to reledmac 2.5.0

It is strongly recommended to stop redefining `\printnpnum` 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.

## 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 edstanzas.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>\@hangingsymbol</code> .....	1
<code>\@iiiminipage</code> .....	1
<code>\@insertstanzaumber</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
<code>CLASSbook</code> .....	295
<code>CLASSmemoir</code> .....	174, 175, 228, 229, 257, 295, 313, 369, 373
<code>CLASSscrbook</code> .....	373
<code>COMMAND\*footnote</code> .....	61
<code>COMMAND\...@footnotemark...</code> .....	177
<code>COMMAND\...d@ta</code> .....	131

COMMAND\<hook	
@<series	217
COMMAND\<hookname	
<pseudoseries	219, 220
COMMAND\<type	
footfmt	165
COMMAND\@@line	158
COMMAND\@MM	146, 370
COMMAND\@Rlineflag	259, 370
COMMAND\@Serefprefix	239
COMMAND\@Serefprefixmore	239
COMMAND\@add@	285
COMMAND\@adv	95
COMMAND\@apprefprefixmore	239
COMMAND\@apprefprefixsingle	239
COMMAND\@bsphack	231
COMMAND\@doclearpage	228, 229, 363, 373
COMMAND\@doreinfeetX	373
COMMAND\@dprintingcolumns	370
COMMAND\@edindex@hyperref	259, 260
COMMAND\@edtext@level	111
COMMAND\@esphack	231
COMMAND\@fnpos	195
COMMAND\@footnotemark	175, 363, 373
COMMAND\@footnotetext	175, 176, 363
COMMAND\@gobble	109
COMMAND\@gobblefive	214, 371
COMMAND\@gobblefour	369
COMMAND\@gobblethree	362
COMMAND\@h	161
COMMAND\@hangingsymbol	263
COMMAND\@iiiminipage	249, 251, 362, 373
COMMAND\@iiiminpage	249
COMMAND\@l	368
COMMAND\@l@dttempcnta	133, 135, 141
COMMAND\@l@dttempcntb	135
COMMAND\@l@reg	368
COMMAND\@lab	92, 230, 233, 236, 362
COMMAND\@ldunboxmpfoot	251
COMMAND\@led@extranofeet	314
COMMAND\@ledinnote@command	255, 256
COMMAND\@lemma	114, 116
COMMAND\@lock	86, 263
COMMAND\@lopL	363
COMMAND\@lopR	363
COMMAND\@makecol	225, 228, 373
COMMAND\@mpfnpos	195
COMMAND\@nl	92, 94–96, 104, 233, 362, 363
COMMAND\@nl@reg	92, 311, 363, 368
COMMAND\@opXfeet	363



COMMAND\@opfeetX	373
COMMAND\@opxtrafeeti	373
COMMAND\@page	94, 233
COMMAND\@pend	363
COMMAND\@pendR	363
COMMAND\@ref	92, 100–102, 105, 109
COMMAND\@ref@reg	101, 363
COMMAND\@reinserts	225, 226, 228, 373
COMMAND\@secondoftwo	61
COMMAND\@sect	298
COMMAND\@series	216
COMMAND\@set	96
COMMAND\@sidenotesepp	248
COMMAND\@sw	102, 117, 120, 121
COMMAND\@tag	111, 112, 115
COMMAND\@tempcnta	73
COMMAND\@tempcntb	73
COMMAND\@toksa	78
COMMAND\@toksb	78
COMMAND\@xloop	142
COMMAND\@xympar	243, 373
COMMAND\Aendnote	14, 23
COMMAND\Afootfmt	145
COMMAND\Afootgroup	145
COMMAND\Afootnote	8, 14, 22, 23, 25, 112, 153, 174, 196, 208, 372
COMMAND\Afootstart	145
COMMAND\AtEveryPend	17, 126, 369, 371, 373
COMMAND\AtEveryPstart	17, 369, 371, 373
COMMAND\Bendnote	14, 22
COMMAND\Bfootnote	8, 14, 174, 196, 208
COMMAND\Centering	38
COMMAND\Cfootnote	174
COMMAND\Columns	73, 150
COMMAND\Dfootnote	174
COMMAND\Efootnote	174
COMMAND\Gls	53
COMMAND\NR	298
COMMAND\Pages	73, 226, 227
COMMAND\ProcessOptionsX	65
COMMAND\RaggedLeft	38
COMMAND\RaggedRight	38
COMMAND\SEonlypage	238, 375
COMMAND\Seref	47–49, 238, 240, 376
COMMAND\Serefonlypage	48, 49, 317, 375
COMMAND\Serefwithpage	47–49, 238, 240, 317, 375
COMMAND\Stanza	368
COMMAND\Waklam	285
COMMAND\X@doreinfeet	226, 373
COMMAND\XXXXXXfmt	310
COMMAND\XXXXXXfmt	310

COMMAND\Xafterlemmaseparator	35, 314
COMMAND\Xafternote	38, 313, 314
COMMAND\Xafternumber	33, 314
COMMAND\Xafterrule	39, 196, 314, 368, 371
COMMAND\Xaftersymlinenum	33, 314
COMMAND\Xarrangement	30, 60, 146, 147, 218, 314
COMMAND\Xarrangement@footparagraph	151
COMMAND\Xarrangement@normal	147
COMMAND\Xarrangement@paragraph	151
COMMAND\Xbeforelemmaseparator	35, 314
COMMAND\Xbeforenotes	39, 196, 314, 368, 371
COMMAND\Xbeforenumber	31, 33, 314
COMMAND\Xbeforesymlinenum	33, 314
COMMAND\Xbhookendnote	315
COMMAND\Xbhookgroup	39, 375
COMMAND\Xbhooknote	37, 315, 316, 373, 374
COMMAND\Xboxendlinenum	34, 315, 372
COMMAND\Xboxlinenum	34, 315
COMMAND\Xboxlinenumalign	34, 315, 372
COMMAND\Xboxstartlinenum	34, 315, 372
COMMAND\Xboxsymlinenum	34, 315
COMMAND\Xcolalign	37, 371
COMMAND\Xdo@feet	225, 363, 373
COMMAND\Xend	214
COMMAND\XendXafterlemmaseparator	315
COMMAND\XendXbeforelemmaseparator	315
COMMAND\XendXinplaceoflemmaseparator	315
COMMAND\XendXlemmaseparator	315
COMMAND\XendXmorethantwolines	315
COMMAND\XendXtwolines	315
COMMAND\Xendafterenumber	33, 374
COMMAND\Xendafterlemmaseparator	36, 315
COMMAND\Xendafternote	40
COMMAND\Xendafternumber	35
COMMAND\Xendafterpagenumber	35, 49
COMMAND\XendafterpagenumberSErefonlypage	317
COMMAND\Xendaftersymlinenum	33, 35, 374
COMMAND\Xendahookinplaceofnumber	35, 374
COMMAND\Xendahooklinenum	35, 374
COMMAND\Xendbeforelemmaseparator	36, 315
COMMAND\Xendbeforelinenum	35
COMMAND\Xendbeforenumber	33, 374
COMMAND\Xendbeforepagenumber	35, 48, 49
COMMAND\XendbeforepagenumberSErefonlypage	48
COMMAND\Xendbeforesymlinenum	33, 35, 374
COMMAND\Xendbhookinplaceofnumber	35, 374
COMMAND\Xendbhooklinenum	35, 374
COMMAND\Xendbhooknote	37
COMMAND\Xendboxendlinenum	34, 372
COMMAND\Xendboxlinenum	34, 315, 370

COMMAND\Xendboxlinenumalign	34, 315, 372
COMMAND\Xendboxstartlinenum	34, 372
COMMAND\Xendboxsymlinenum	34, 374
COMMAND\Xendhangindent	37, 374
COMMAND\Xendinplaceoflemmaseparator	36, 315
COMMAND\Xendinplaceofnumber	34, 373
COMMAND\Xendinsertsep@	202
COMMAND\Xendlemmadisablefontselection	36
COMMAND\Xendlemmafont	36, 375
COMMAND\Xendlemmaseparator	24, 36, 315
COMMAND\Xendletboxendlinenum	315
COMMAND\Xendlineflag	49
COMMAND\Xendlineprefixmore	35, 49
COMMAND\Xendlineprefixsingle	35, 49
COMMAND\Xendlinangeseparator	31, 49, 317, 374
COMMAND\Xendmorethantwolines	23, 32, 49, 315, 371, 372
COMMAND\Xendnonumber	32, 373
COMMAND\Xendnote	199, 213, 214, 371
COMMAND\Xendnotefontsize	37, 314
COMMAND\Xendnotenumfont	35, 36, 314
COMMAND\Xendnumberonlyfirstinline	31, 374
COMMAND\Xendnumberonlyfirstintwolines	31, 374
COMMAND\Xendparagraph	40, 368
COMMAND\Xendsep	40
COMMAND\Xendsublinesep	33, 49
COMMAND\Xendsymlinenum	31, 374
COMMAND\Xendtwolines	23, 32, 49, 315, 371, 372
COMMAND\Xendtwolinesbutnotmore	32, 49, 371, 372
COMMAND\Xendtwolinesonlyinsamepage	32, 49, 371, 372
COMMAND\Xfootfmt	316
COMMAND\Xfootgroup	150
COMMAND\Xfootins	149
COMMAND\Xfootnote	46, 51, 111, 310, 365, 369–371, 375
COMMAND\Xfootstarts	150
COMMAND\Xhangindent	37, 374
COMMAND\Xhsize	40
COMMAND\Xhsizethreecol	38, 40, 315
COMMAND\Xhsizetwocol	38, 40, 219, 315
COMMAND\Xinplaceoflemmaseparator	35, 315
COMMAND\Xinplaceofnumber	33, 315, 370, 372
COMMAND\Xinsertparafootsep	156, 158
COMMAND\Xledsetnormalparstuff	312, 313, 371
COMMAND\Xlemmadisablefontselection	36
COMMAND\Xlemmafont	36, 375
COMMAND\Xlemmaseparator	35, 220, 222, 225, 315
COMMAND\Xlineflag	49
COMMAND\Xlinangeseparator	31, 49, 317, 374
COMMAND\Xmaxhnotes	39, 60, 314, 315, 368, 370
COMMAND\Xmorethantwolines	23, 31, 32, 49, 315, 370
COMMAND\Xnoindent	374

COMMAND\Xnolemmaseparator	35, 225, 315
COMMAND\Xnonbreakableafternumber	33, 315, 366
COMMAND\Xnonumber	32, 315
COMMAND\Xnonumberinfootnote	315
COMMAND\Xnotefontsize	37, 314
COMMAND\Xnotefontsize@⟨s⟩	156, 160, 161
COMMAND\Xnotenumfont	36, 314
COMMAND\Xnoteswidthliketwocolumns	40, 369
COMMAND\Xnumberonlyfirstinline	31, 89, 219, 220, 222, 315, 365, 370
COMMAND\Xnumberonlyfirstintwolines	31, 315, 365
COMMAND\Xonlypstart	32, 315, 365, 370
COMMAND\Xparafootsep	38, 89, 314, 315
COMMAND\Xparafootsep@series	156
COMMAND\Xparindent	37, 371, 374
COMMAND\Xpstart	32, 312, 315, 365, 370
COMMAND\Xpstarteverytime	32, 312, 315, 370
COMMAND\Xragged	38
COMMAND\Xrigidbalance	158, 317, 375
COMMAND\Xstanza	33, 44
COMMAND\Xstanzaseparator	33
COMMAND\Xsublinesep	20, 33, 49, 317
COMMAND\Xsublinesepside	20, 33
COMMAND\Xsymlinenum	31, 38, 314, 315, 372
COMMAND\Xtwolines	23, 31, 32, 49, 170, 171, 219, 315, 370
COMMAND\Xtwolinesappref	219
COMMAND\Xtwolinesbutnotmore	32, 49, 315, 371
COMMAND\Xtwolinesbutnotmoreappref	220
COMMAND\Xtwolinesonlyinsamepage	32, 49, 315, 371
COMMAND\Xtxtbeforenotes	39, 315
COMMAND\Xunvxh	154, 316
COMMAND\&	314
COMMAND\absline@num	86, 132
COMMAND\accent	110
COMMAND\actionlines@list	87, 134
COMMAND\actions@list	87
COMMAND\add@inserts	87, 140
COMMAND\add@inserts@next	140, 141
COMMAND\add@penalties	132, 141
COMMAND\addcontentsline	292
COMMAND\addfootins	310, 313
COMMAND\addfootinsX	310, 313
COMMAND\advancelabel@refs	232
COMMAND\advanceline	20, 21, 88, 95, 106, 373
COMMAND\advancepageno	225
COMMAND\affixlin@num	248
COMMAND\affixline@num	135, 137, 139, 363
COMMAND\affixpstart@num	139
COMMAND\afterXrule	314
COMMAND\afterXsymlinenum	314
COMMAND\afterenumber	33

COMMAND\aftergroup	109, 113
COMMAND\afterlemmaseparator	314
COMMAND\afternote	314
COMMAND\afternoteX	38, 313, 314
COMMAND\afternumberinfootnote	314
COMMAND\afterruleX	39, 368, 371
COMMAND\applabel	48, 233, 234, 371
COMMAND\appref	46, 48, 49, 238, 240, 375, 376
COMMAND\apprefprefixmore	48, 314
COMMAND\apprefprefixsingle	48, 314
COMMAND\apprefwithpage	48, 49, 238, 240, 372, 375
COMMAND\arrangementX	30, 60, 178, 218, 314
COMMAND\arrangementX@normal	183
COMMAND\at@every@pend	126
COMMAND\autopar	16, 123, 127, 128, 193, 364, 366, 367, 371
COMMAND\ballast	60
COMMAND\ballast@count	132, 141
COMMAND\baselineskip	30, 152, 156
COMMAND\beforeXnotes	314
COMMAND\beforeXsymlinenum	314
COMMAND\beforeeledchapter	9, 57, 293, 294
COMMAND\beforelemmaseparator	314
COMMAND\beforenotesX	39, 316, 367, 368, 371
COMMAND\beforenumberinfootnote	314
COMMAND\begin	270
COMMAND\beginnumbering	15, 17, 18, 74, 77, 85, 89, 90, 103, 127, 198, 365, 368, 372, 373
COMMAND\bf	365
COMMAND\bfseries	36, 365
COMMAND\bhookXnote	315
COMMAND\bhookgroupX	39, 375
COMMAND\bhooknoteX	37, 316, 373, 374
COMMAND\body	264
COMMAND\bodyfootmarkA	28
COMMAND\boxXendlinenum	315
COMMAND\boxXendlinenumalign	315
COMMAND\boxXendstartlinenum	315
COMMAND\boxendlinenum	315
COMMAND\boxlinefootnote	167
COMMAND\boxlinenum	315
COMMAND\boxlinenumalign	315
COMMAND\boxstartlinenum	315
COMMAND\boxsymlinenum	315
COMMAND\break	30, 154
COMMAND\brokenpenalty	141
COMMAND\centering	38
COMMAND\ch@ck@l@ck	364
COMMAND\ch@cksub@l@ck	137, 364
COMMAND\chapter	57, 295, 368, 371, 373
COMMAND\chaptermark	293
COMMAND\check@pb@in@verse	306

COMMAND\colalignX	37, 371
COMMAND\collect@body	271
COMMAND\colorbox	61
COMMAND\columns	40
COMMAND\columnwidth	152, 369
COMMAND\command names	219
COMMAND\copyright	110
COMMAND\correct@Xfootins@box	370
COMMAND\correct@footinsX@box	370
COMMAND\count	159, 160
COMMAND\critex	364
COMMAND\critext	116, 309, 310, 313
COMMAND\csname	65, 119
COMMAND\ctab	286, 291
COMMAND\ctabtext	291
COMMAND\dcoll	280
COMMAND\def	63
COMMAND\detokenize	119
COMMAND\dimen	160
COMMAND\dimexpr	40
COMMAND\discretionary	153
COMMAND\displaywidowpenalty	141
COMMAND\do@actions	132–134, 364
COMMAND\do@actions@fixedcode	363
COMMAND\do@actions@next	133, 134
COMMAND\do@ballast	132, 141
COMMAND\do@feetX	373
COMMAND\do@insidelinehook	366
COMMAND\do@line	87, 108, 125, 128, 131, 140, 141, 263, 364, 366, 368
COMMAND\do@linehook	364
COMMAND\do@lockoff	88
COMMAND\do@lockon	88
COMMAND\dodoreintrafeet	362
COMMAND\doendnotes	23, 202, 316, 371
COMMAND\doendnotesbysection	24, 202, 214, 372
COMMAND\doinsidelinehook	21, 369
COMMAND\dolinehook	21, 369
COMMAND\doreintrafeeti	373
COMMAND\doreintrafeetii	373
COMMAND\doxtrafeet	225, 362
COMMAND\doxtrafeeti	373
COMMAND\doxtrafeetii	373
COMMAND\dummy@ref	109
COMMAND\edaftertab	56, 286, 287
COMMAND\edatleft	55, 284
COMMAND\edatright	55, 56, 284
COMMAND\edbforetab	56, 286
COMMAND\edfilldimen	285
COMMAND\edfont@info	115
COMMAND\edgls	53, 254

COMMAND\edindex	51–53, 254, 255, 258, 260, 274, 366, 369, 370, 373, 374
COMMAND\edindexlab	53
COMMAND\edlabel	45–48, 110, 230, 232, 233, 235, 236, 242, 254, 274, 362, 365–367, 370, 375
COMMAND\edlabelE	47, 234
COMMAND\edlabelS	47, 234
COMMAND\edlabelSE	47
COMMAND\edlineref	46, 230, 315, 370, 372, 375
COMMAND\edmakelabel	47, 242
COMMAND\edpageref	46, 230, 235, 242
COMMAND\edrowfill	285
COMMAND\edtabcolsep	279
COMMAND\edtext	6, 22–28, 41, 46–48, 51, 54, 61, 86, 87, 100–102, 105, 108–116, 118–122, 233, 234, 237, 274, 276, 295, 309, 310, 313, 363, 364, 366, 368–372
COMMAND\edtext@level	372
COMMAND\edvertdots	56, 284
COMMAND\edvertline	56, 284
COMMAND\elechapter	57
COMMAND\eled@sectioning@out	299
COMMAND\eledchapter	57, 313, 369, 373
COMMAND\eledchapter*	57
COMMAND\eledmac@error	362
COMMAND\eledsection	6, 14, 57, 109, 130, 293, 313, 370
COMMAND\eledsection*	57
COMMAND\eledsubsection	57, 313
COMMAND\eledsubsection*	57
COMMAND\eledsubsubsection	57, 313
COMMAND\eledsubsubsection*	57
COMMAND\eledxxx	9, 58, 294, 299, 368
COMMAND\eledxxxx	293
COMMAND\else	253, 293
COMMAND\empty	72, 136, 230
COMMAND\end	270
COMMAND\end@lemmas	109
COMMAND\endashchar	41, 164
COMMAND\endgraf	125, 155, 193
COMMAND\endlock	20, 88, 107, 268
COMMAND\endminipage	249, 251, 362, 373
COMMAND\endnotes	371, 375, 376
COMMAND\endnumbering	15, 18, 74, 76, 77, 364, 372
COMMAND\endprint	199, 201, 214, 312
COMMAND\endstanzaextra	314
COMMAND\endsub	20, 88, 105
COMMAND\endverse	316
COMMAND\everypar	127
COMMAND\extensionchars	59, 74
COMMAND\f@x@l@cks	364
COMMAND>falseverse	313, 366, 368
COMMAND\fi	293
COMMAND\firstlinenum	18, 135, 364
COMMAND\firstsublinenum	18, 364

COMMAND\fix@page	92, 94, 363
COMMAND\flag@end	105, 115, 368
COMMAND\flag@start	105, 115, 368, 369
COMMAND\flagstanza	44
COMMAND\floatingpenalty	146, 370
COMMAND\flush@notes	142
COMMAND\fnpos	195, 367
COMMAND\footfmt	145, 148
COMMAND\footfmt...	179
COMMAND\footfootmarkA	28
COMMAND\footfudgefactor	154
COMMAND\footfudgefiddle	60, 152, 362
COMMAND\footgroup	145
COMMAND\footins	149
COMMAND\footnormal	218, 314, 363
COMMAND\footnormalX	314
COMMAND\footnote	28, 60, 174–176, 311, 363
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	224
COMMAND\footnotelang@lua	144
COMMAND\footnotelang@poly	144
COMMAND\footnoteoption@	143, 374
COMMAND\footnoterule	159
COMMAND\footnotesize	37
COMMAND\footparagraph	152, 218, 314, 368
COMMAND\footparagraphX	188, 314, 368
COMMAND\footplitskips	364, 370
COMMAND\footstart	145, 149, 159
COMMAND\footstrut	156
COMMAND\footthreecol	314
COMMAND\footthreecolX	314, 371
COMMAND\foottwocol	314
COMMAND\foottwocolX	314, 371
COMMAND\fullstop	41, 317
COMMAND\get@edindex@hyperref	259
COMMAND\get@edindex@ledinnote@command	255
COMMAND\get@fnmark	176
COMMAND\get@index@command	367
COMMAND\get@linelistfile	364
COMMAND\get@thisfootnote	182
COMMAND\getline@num	132, 133
COMMAND\gl@p	78
COMMAND\global	92
COMMAND\globaldefs	92



COMMAND\gls	53, 262
COMMAND\hangindentX	37, 371, 374
COMMAND\hangingsymbol	314, 364
COMMAND\hbox	153
COMMAND\hfill	367
COMMAND\hidenumbering	21, 100, 371
COMMAND\hline	54
COMMAND\hrulefill	285
COMMAND\hsize	30, 149, 152, 154, 160, 162, 194, 363, 369
COMMAND\hsizeX	40
COMMAND\hsizethreecol	315
COMMAND\hsizethreecolX	38, 40
COMMAND\hsizetwocol	315
COMMAND\hsizetwocolX	38, 40
COMMAND\hyperlinkR	258
COMMAND\hyperlinkformat	258
COMMAND\hyperlinkformatR	259
COMMAND\if@RTL	67
COMMAND\if@edtext@	369, 372
COMMAND\if@eled@sectioning	294
COMMAND\if@noneed@Footnote	105
COMMAND\ifXnote@	73
COMMAND\ifbypage@	79
COMMAND\ifbypage@R	79
COMMAND\ifbypstart@	79
COMMAND\ifbypstart@R	79
COMMAND\iffirst@linenum@out@	103, 104
COMMAND\ifindtl@innote	73
COMMAND\ifindtl@notenumber	73
COMMAND\ifinserthangingsymbol	263
COMMAND\ifinstanza	263
COMMAND\ifstwofollowinglines	171
COMMAND\ifl@d@Xmorethantwolines	169, 371
COMMAND\ifl@d@Xtwolines	169
COMMAND\ifl@d@dash	169
COMMAND\ifl@d@elin	168
COMMAND\ifl@d@esl	169
COMMAND\ifl@d@pnum	168
COMMAND\ifl@d@ssub	168
COMMAND\ifl@dend@X	213
COMMAND\ifl@dmemoir	362
COMMAND\ifl@dpaging	369
COMMAND\ifl@dpairing	73, 364
COMMAND\ifl@dprintingpages	370
COMMAND\ifl@dskipnumber	135
COMMAND\ifl@dstartendok	285
COMMAND\ifl@imakeidx	67
COMMAND\ifledRcol	73, 365
COMMAND\ifledRcol@	73, 368
COMMAND\iflemmacommand@	370

COMMAND\ifnoend@	202
COMMAND\ifnoledgroup@	253
COMMAND\ifnoteschanged@	89
COMMAND\ifnumberedpar@	123
COMMAND\ifnumbering	74, 77
COMMAND\ifnumberingR	73, 365
COMMAND\ifnumberline	114, 135
COMMAND\ifpst@rted	364
COMMAND\ifpst@rtedL	74
COMMAND\ifseriesbefore	216
COMMAND\ifsublines@	86, 97
COMMAND\iftrue	372
COMMAND\ifvmode	232
COMMAND\ifxxx	293
COMMAND\ignorespaces	112
COMMAND\imki@wrindexentry	67
COMMAND\immediate	103, 104, 198
COMMAND\indent	17, 127, 316
COMMAND\index	261
COMMAND\indtl@wrindexentry	67
COMMAND\initnumbering@quote	291, 373
COMMAND\initnumbering@reg	364
COMMAND\initnumbering@sectcmd	373
COMMAND\inplaceoflemmaseparator	315
COMMAND\inplaceofnumber	315
COMMAND\insert	140, 145, 148, 179
COMMAND\insert@count	100, 105, 112
COMMAND\insert@countR	112
COMMAND\insertthangingsymbol	367
COMMAND\insertlines@list	87, 100, 101
COMMAND\insertparafootsepX	192
COMMAND\inserts@list	108, 123, 140, 153
COMMAND\interAfootnotelinepenalty	363
COMMAND\interfootnotelinepenalty	363
COMMAND\interlinepenalty	145
COMMAND\interparanoteglue	313
COMMAND\justifying	38
COMMAND\l@advance@parledegroupp@beforenormalnotes	373
COMMAND\l@d@@wrindexhyp	369
COMMAND\l@d@add	117
COMMAND\l@d@end	198, 213
COMMAND\l@d@nums	112, 114, 116, 117, 168
COMMAND\l@d@section	199
COMMAND\l@d@set	96, 107
COMMAND\l@dampcount	276
COMMAND\l@dbfnote	176, 363
COMMAND\l@dcheckstartend	285
COMMAND\l@dchset@num	96
COMMAND\l@dcolcount	276, 277
COMMAND\l@dcollect@@body	270

COMMAND\l@dcollect@body	270
COMMAND\l@dcsnote	368
COMMAND\l@dcsnotetext	131, 246, 247
COMMAND\l@dcsnotetext@l	131, 246, 247
COMMAND\l@dcsnotetext@r	131, 246
COMMAND\l@ddodoreintrafeet	226, 362
COMMAND\l@ddoxtrafeet	226, 362
COMMAND\l@demptyd@ta	364
COMMAND\l@dend@close	198
COMMAND\l@dend@open	198
COMMAND\l@dend@stuff	198
COMMAND\l@denvbody	270
COMMAND\l@dfeetbeginmini	363
COMMAND\l@dfeetendmini	363
COMMAND\l@dgetline@margin	364
COMMAND\l@dgetlock@disp	364
COMMAND\l@dgetref@num	236, 237
COMMAND\l@dgetsidenote@margin	243, 364
COMMAND\l@dgobbeloptarg	369
COMMAND\l@dgobblearg	369
COMMAND\l@dgobbleoptarg	275
COMMAND\l@dlabel@parse	236, 237
COMMAND\l@dld@ta	135, 137
COMMAND\l@dlp@rbox	247
COMMAND\l@dlsn@te	364
COMMAND\l@dlsnote	368
COMMAND\l@dmake@labels	232
COMMAND\l@dnumpsstartL	74, 364
COMMAND\l@dp@rsefootspec	169
COMMAND\l@dpush@begins	270
COMMAND\l@drd@ta	135, 137
COMMAND\l@dref@undefined	236
COMMAND\l@drrn@te	364
COMMAND\l@drrsnote	368
COMMAND\l@dtabaddcols	285
COMMAND\l@dtabnoexpands	362
COMMAND\l@dumboxmpfoot	373
COMMAND\l@dunboxmpfoot	364
COMMAND\l@dzeropenalties	364, 369
COMMAND\l@pb	305
COMMAND\l@prev@nopb	304
COMMAND\l@prev@pb	304
COMMAND\l@reg	311
COMMAND\label	17, 47, 53, 231, 237
COMMAND\label@refs	230
COMMAND\labelstarttrue	17, 365
COMMAND\labelref@list	230, 233
COMMAND\language	153
COMMAND\last@page@num	363
COMMAND\lastbox	127

COMMAND\lastskip	106
COMMAND\leavevmode	17, 127
COMMAND\led@check@nopb	305
COMMAND\led@check@pb	305
COMMAND\led@nopb	304–306
COMMAND\led@nopbnum	304, 305
COMMAND\led@pb	304–306
COMMAND\led@pb@macro	305
COMMAND\led@pbnum	304, 305
COMMAND\led@reinit@index@fornote	261
COMMAND\led@set@index@fornote	261
COMMAND\ledRflag	258
COMMAND\ledchapter	313, 366
COMMAND\ledfootinsdim	314
COMMAND\ledinnernote	50, 245, 368
COMMAND\ledinnote	257, 312, 372
COMMAND\ledinnotemark	51, 312, 371
COMMAND\ledleftnote	50, 245
COMMAND\ledlinenum	84, 364
COMMAND\ledllfill	131
COMMAND\ledlsnotesep	50
COMMAND\ledlsnotewidth	50
COMMAND\lednopb	58, 304
COMMAND\lednopbinverse	306
COMMAND\lednopbinversetrue	44, 58
COMMAND\lednopbnum	304
COMMAND\ledouternote	50, 245, 368
COMMAND\ledpb	58, 304, 313
COMMAND\ledpbnum	304
COMMAND\ledpbsetting	59, 305, 374
COMMAND\ledrightnote	50, 245
COMMAND\ledrsnotesep	50
COMMAND\ledrsnotewidth	50
COMMAND\ledsection	313
COMMAND\ledsectnomark	293
COMMAND\ledsectnotoc	292
COMMAND\ledsetnormalparstuff	312, 313, 371
COMMAND\ledsetnormalparstuff@common	193
COMMAND\ledsetnormalparstuffX	312, 313, 371
COMMAND\ledsidenote	50, 245–247
COMMAND\ledsubsection	313
COMMAND\ledsubsubsection	313
COMMAND\ledxxx	368
COMMAND\left	55
COMMAND\leftctab	286
COMMAND\leftheadline	84
COMMAND\leftlinenum	19, 84, 362, 364
COMMAND\leftltab	286
COMMAND\leftnoteupfalse	50
COMMAND\leftpstartnum	139

COMMAND\lefttab	286
COMMAND\leftsidenote	246
COMMAND\leftskip	149, 152, 154
COMMAND\lemma	2, 22, 24–28, 108, 111–113, 115, 116, 118, 309, 364, 365, 372, 373, 375
COMMAND\lemmaseparator	315
COMMAND\let	25, 41, 268, 362
COMMAND\letboxXendendlinum	315
COMMAND\line	158, 161
COMMAND\line@list	86, 102, 115
COMMAND\line@list@stuff	74, 89, 90, 103, 362, 364
COMMAND\line@list@version	92
COMMAND\line@margin	80, 137, 243
COMMAND\line@num	85, 86, 88, 135, 362
COMMAND\line@set	116, 117
COMMAND\lineation	19, 79
COMMAND\linebreak	30
COMMAND\linenum	22, 24, 25, 46–48, 108, 116, 235, 237, 243, 309
COMMAND\linenum@out	103, 230, 233
COMMAND\linenumberlist	18, 19, 72, 136, 362
COMMAND\linenumberstyle	21, 84, 362
COMMAND\linenumincrement	18, 364
COMMAND\linenummargin	19, 80, 243
COMMAND\linenumr@p	84, 362, 364
COMMAND\linenumrep	84, 364
COMMAND\linenumsep	19, 50, 84, 244
COMMAND\linrangesep@	224
COMMAND\lineref	230, 235, 242, 315, 370
COMMAND\list@clear	78
COMMAND\list@clearing@reg	364
COMMAND\list@create	78
COMMAND\lock@disp	83
COMMAND\lock@off	99
COMMAND\lock@on	98
COMMAND\lockdisp	20, 83
COMMAND\loop	142, 143, 264
COMMAND\ltab	286, 287, 291
COMMAND\ltabtext	291
COMMAND\m@mmf@prepare	175
COMMAND\makeatletter	131
COMMAND\makehboxoffhboxes	155, 156
COMMAND\makeindex	51, 258
COMMAND\makelabel	242
COMMAND\managestanza@modulo	265
COMMAND\marginpar	50, 60, 243, 363
COMMAND\marginparwidth	50, 244
COMMAND\markboth	131
COMMAND\mathchardef	264
COMMAND\maxhXnotes	315
COMMAND\maxhnotesX	40, 60, 314, 367, 368, 370, 371
COMMAND\maxlinesinpar@list	90

COMMAND\measurebody	288
COMMAND\measuretbody	288
COMMAND\memorybreak	18
COMMAND\morenoexpands	61, 108, 110
COMMAND\morethantwolines	315
COMMAND\mpfnpos	195, 367
COMMAND\mpnormalfootgroup	363
COMMAND\mpnormalvfootnote	363
COMMAND\multfootsep	28, 174
COMMAND\multiplefootnotemark	174
COMMAND\musixtex	368
COMMAND\n@num	364, 371
COMMAND\n@num@ref	371
COMMAND\new@line	104, 363
COMMAND\newcommand	25, 63, 174, 232
COMMAND\newcommandx	25
COMMAND\newhookarg@specific	224
COMMAND\newhookcommand@series	219, 220, 371
COMMAND\newhookcommand@series@reload	220
COMMAND\newhookcommand@toggle@reload	220, 369
COMMAND\newhooktoggle@series	219, 220, 371
COMMAND\newhooktoggle@specific	224
COMMAND\newif	371
COMMAND\newline	30
COMMAND\newlinechar	213
COMMAND\newseries	29, 310, 313, 314
COMMAND\newseries@	206, 207, 217
COMMAND\newverse	44, 45, 313, 368
COMMAND\next	264
COMMAND\next@action	90
COMMAND\next@actionline	90
COMMAND\next@insert	141
COMMAND\nl@regR	92
COMMAND\no@expands	61, 115, 362
COMMAND\noXlemmaseparator	315
COMMAND\nobreak	168
COMMAND\nocritical	207
COMMAND\noeledsec	58, 313
COMMAND\noendnotes	313
COMMAND\noexpand	311
COMMAND\nofamiliar	222
COMMAND\noindent	17, 127, 374
COMMAND\noindentX	374
COMMAND\nomk@	224
COMMAND\nonbreakableafternumber	315
COMMAND\nonnumberinfootnote	315
COMMAND\normal@footnotemarkX	178
COMMAND\normal@page@break	304
COMMAND\normal@pars	193
COMMAND\normalbfnoteX	364

COMMAND\normalbodyfootmarkX	178
COMMAND\normalfootfmt	41, 148, 155, 164, 199
COMMAND\normalfootfmtX	179, 180
COMMAND\normalfootfootmarkX	180
COMMAND\normalfootgroup	150
COMMAND\normalfootgroupX	180
COMMAND\normalfootnoterule	146
COMMAND\normalfootstart	149, 152
COMMAND\normalfootstartX	180
COMMAND\normalvfootnote	148
COMMAND\normalvfootnoteX	179
COMMAND\notbool	293
COMMAND\notfontsetup	314
COMMAND\notfontsizeX	37, 314
COMMAND\notenumfont	314
COMMAND\notenumfontX	36, 314
COMMAND\notesXwidthliketwocolumns	315
COMMAND\noteswidthliketwocolumnsX	40, 315, 369, 371
COMMAND\num@lines	123, 141
COMMAND\numberlinefalse	18
COMMAND\numberlinetrue	18
COMMAND\numberonlyfirstinline	217, 315
COMMAND\numberonlyfirstintwolines	315
COMMAND\numberpstartfalse	17
COMMAND\numberpstarttrue	17, 32, 312, 364, 373
COMMAND\numberstanza	33
COMMAND\numberstanzafalse	44
COMMAND\numberstanzatrue	44
COMMAND\numlabfont	19, 41, 84
COMMAND\one@line	123
COMMAND\onehalfspacing	374
COMMAND\onlyXpstart	315
COMMAND\page@action	87, 97
COMMAND\page@start	87, 364
COMMAND\pagecontents	87
COMMAND\pagelinesep	52
COMMAND\pageno	225
COMMAND\pageparbreak	313
COMMAND\pageref	47, 235
COMMAND\par	24, 30, 127, 193
COMMAND\par@line	123, 141
COMMAND\para@footgroup	152
COMMAND\para@footgroupX	191
COMMAND\para@footsetup	152, 362
COMMAND\para@footsetupX	189, 362, 369
COMMAND\para@vfootnote	156
COMMAND\para@vfootnoteX	190
COMMAND\parafootfmt	155
COMMAND\parafootfmtX	191
COMMAND\parafootftm	158

COMMAND\parafootftmX	192
COMMAND\parafootftmsep	314
COMMAND\parafootsep	315, 367, 372
COMMAND\parafootsepX	38, 89, 314, 315
COMMAND\parafootstart	152
COMMAND\parafootstartX	189
COMMAND\paravfootnote	153
COMMAND\parfillskip	155
COMMAND\parindent	374
COMMAND\parindentX	37, 374
COMMAND\parshape	60
COMMAND\parskip	127
COMMAND\pausenumbering	18, 77, 90, 91, 128, 367, 369
COMMAND\penalty	155
COMMAND\pend	2, 6, 16–19, 21, 57, 58, 106, 108, 111, 117, 123–128, 139, 140, 311, 367, 368
COMMAND\preXnotes	39, 196, 371
COMMAND\preXnotes@	149, 196, 365
COMMAND\prenotesX	39, 197, 371
COMMAND\prepare@preXnotes	196
COMMAND\prev@nopb	305
COMMAND\prev@pb	305
COMMAND\prevlineX	89
COMMAND\prevpageX@num	89
COMMAND\print@Xfootnoterule	372
COMMAND\print@Xnotes	226, 227
COMMAND\print@Xnotes@forpages	370
COMMAND\print@eledsection	130
COMMAND\print@footnoteXrule	372
COMMAND\print@leftmargin@eledsection	294
COMMAND\print@line	129
COMMAND\print@notesX@forpages	370
COMMAND\print@rightmargin@eledsection	294
COMMAND\printendlines	203, 240, 317, 362, 364
COMMAND\printlinefootnote	165, 167, 370
COMMAND\printlinefootnotearea	167, 168, 370
COMMAND\printlinefootnotenumbers	165
COMMAND\printlines	148, 164, 168, 169, 203, 240, 317, 362, 364, 371, 375
COMMAND\printnpnum	312, 317
COMMAND\printpstart	165
COMMAND\protect	110, 311
COMMAND\providecommand	174, 362
COMMAND\pstart	2, 6, 16–19, 21, 57, 58, 96, 106, 107, 111, 117, 123–127, 130, 140, 311, 364, 365, 367–369, 371–373
COMMAND\pstartinfootnote	315
COMMAND\pstartinfootnoteeverytime	315
COMMAND\pstartnum	139
COMMAND\pstartref	46, 230, 236, 367
COMMAND\pstarts	365
COMMAND\raggedX	38
COMMAND\raggedleft	38



COMMAND\raggedright	38
COMMAND\raw@text	123
COMMAND\rbracket	35, 36, 41
COMMAND\read@linelist	89, 91
COMMAND\ref	47, 53
COMMAND\reformatted@	240
COMMAND\reformattedwithpage	240
COMMAND\relax	17, 96, 133, 140, 268, 275, 311
COMMAND\renewcommand	60, 314, 317
COMMAND\resetprevline@	89
COMMAND\resetprevpage@	89
COMMAND\resumenumbering	18, 74, 77, 90, 91, 128, 364, 368, 369
COMMAND\right	55
COMMAND\rightctab	286
COMMAND\rightlinenum	19, 84, 362, 364
COMMAND\rightltab	287
COMMAND\rightnoteupfalse	50
COMMAND\rightrtab	287
COMMAND\rightright	246
COMMAND\rightright	149, 152, 154, 155
COMMAND\rightright	139
COMMAND\rightright	158, 159, 161, 317, 375
COMMAND\rightrightX	158, 317, 375
COMMAND\robustify	31
COMMAND\roman	280, 375
COMMAND\rtab	286, 287, 291
COMMAND\rtabtext	288, 291
COMMAND\sameword	26–28, 117–119, 121, 370, 372, 374
COMMAND\sameword@inedtext	118
COMMAND\saweword	118
COMMAND\scriptsize	84
COMMAND\section	57, 364
COMMAND\section@num	74
COMMAND\sectionmark	293
COMMAND\select@lemmafont	41, 143
COMMAND\series	206
COMMAND\series@	206
COMMAND\seriesatbegin	29, 216, 371
COMMAND\seriesatend	29, 216, 372
COMMAND\set@line	114
COMMAND\set@line@action	88, 97
COMMAND\setSErefonlypageprefixmore	48, 240, 375
COMMAND\setSErefonlypageprefixsingle	48, 240, 375
COMMAND\setSErefprefixmore	48
COMMAND\setSErefprefixsingle	48
COMMAND\setapprefprefixmore	48, 314
COMMAND\setapprefprefixsingle	48, 314, 375
COMMAND\setcommand@series	218
COMMAND\sethangingsymbol	43, 263, 314, 374
COMMAND\sethangingsymbol	42

COMMAND\setistwofollowinglines	171
COMMAND\setl@dlprbox	247
COMMAND\setline	20, 21, 88, 92, 96, 106, 110, 125, 373
COMMAND\setlinenum	21, 92, 96, 107, 362
COMMAND\setprintendlines	203, 204, 364
COMMAND\setprintlines	169, 171, 203, 364
COMMAND\setsidenoteseq	51
COMMAND\setsidenotsep	314
COMMAND\setstanzaindent	265
COMMAND\setstanzaindents	43, 264, 311
COMMAND\setstanzapenalties	264
COMMAND\setstanzavalues	264
COMMAND\settoggle@series	217, 365, 369
COMMAND\showlemma	109, 363
COMMAND\showwordrank	28, 119
COMMAND\sidenote@margin	363
COMMAND\sidenotemargin	50, 363, 368
COMMAND\sidenoteseq	314
COMMAND\sidepstartnumtrue	17
COMMAND\skip	149
COMMAND\skipnumbering	21, 99, 100, 107, 364, 372
COMMAND\skipnumbering@reg	372
COMMAND\small	37
COMMAND\special	12
COMMAND\splitmaxdepth	146, 160
COMMAND\splitoff	158
COMMAND\splittopskip	145, 160, 161
COMMAND\stanza	20, 21, 44, 45, 267, 314, 316, 374
COMMAND\stanza@hang	267
COMMAND\stanza@line	267
COMMAND\stanzaindent	43, 265, 370
COMMAND\stanzaindent*	43
COMMAND\stanzaindentbase	264
COMMAND\stanzanumwrapper	44
COMMAND\startlock	20, 88, 107, 268
COMMAND\startstanzahook	314
COMMAND\startsub	20, 88, 105
COMMAND\strip@pt	152
COMMAND\strutbox	160
COMMAND\sub@action	88, 97
COMMAND\sub@lock	86
COMMAND\sub@off	95, 233
COMMAND\sub@on	95, 233
COMMAND\subline@num	85, 86, 88
COMMAND\sublinenum@rep	362
COMMAND\sublinenumberstyle	21, 84, 362
COMMAND\sublinenumincrement	18
COMMAND\sublinenumr@p	84, 362, 364
COMMAND\sublinenumrep	84, 364
COMMAND\sublineref	46, 230, 235

COMMAND\subsectionmark	293
COMMAND\sw@inthisedtext	112
COMMAND\sw@list@inedtext	115, 122
COMMAND\symlinenum	315
COMMAND\symplinenum	314
COMMAND\sza@penalty	267
COMMAND>tag	370
COMMAND\text	309
COMMAND\textcolor	61
COMMAND\textheight	60
COMMAND\the	362
COMMAND\thefootnoteA	28
COMMAND\thefootnoteX	366
COMMAND\thelabidx	260
COMMAND\thepage	92
COMMAND\thepstart	17
COMMAND\thepstartL	365
COMMAND\thepstartR	365
COMMAND\thestanza	44
COMMAND\this@line@list@version	103
COMMAND\thisfootnote	182
COMMAND\threecolfootfmt	160
COMMAND\threecolfootfmtX	187
COMMAND\threecolfootgroup	159
COMMAND\threecolfootgroupX	187
COMMAND\threecolfootsetup	159
COMMAND\threecolfootsetupX	186
COMMAND\threecolvfootnote	160
COMMAND\threecolvfootnoteX	187
COMMAND\twocolfootfmtX	185
COMMAND\twocolfootgroupX	185
COMMAND\twocolfootsetupX	184
COMMAND\twocolvfootnoteX	185
COMMAND\twolines	217, 315
COMMAND\twolines@A	217
COMMAND\twolines@B	217
COMMAND\twolines@C	217
COMMAND\twolinesbutnotmore	315
COMMAND\twolinesonlyinsamepage	315
COMMAND\txtbeforeXnotes	315
COMMAND\unhbox	153
COMMAND\unpenalty	155, 156
COMMAND\unskip	155
COMMAND\unvxh	155, 316
COMMAND\unvxhX	316
COMMAND\upbracefill	285
COMMAND\usingcritext	310, 313
COMMAND\usingedtext	310, 313
COMMAND\vAfootnote	145
COMMAND\variant	25

COMMAND\ vbox	125, 127, 153, 154, 158, 195
COMMAND\ vfootnote	145, 149, 153, 160
COMMAND\ vl@dbfnote	176, 363
COMMAND\ vnumfootnoteX	364
COMMAND\ vsize	39, 40, 60
COMMAND\ vsplit	141
COMMAND\ wklam	285
COMMAND\ wklamec	285
COMMAND\ wapunktel	285
COMMAND\ wastricht	285
COMMAND\ wrap@edcrossref	235, 369
COMMAND\ wrapped@bodyfootmarkX	192
COMMAND\ wrapped@footfootmarkX	192
COMMAND\ x...	46
COMMAND\ xdef	78, 268
COMMAND\ xflagref	47, 236, 317, 375
COMMAND\ xleft@appenditem	78, 109
COMMAND\ xlineref	46, 47, 317, 375
COMMAND\ xpageref	46
COMMAND\ xpstartref	46, 367
COMMAND\ xright@appenditem	78
COMMAND\ xsublineref	46
COMMAND\ xxref	47, 237, 243, 367, 370, 371
COMMAND\ zz@@@	362
ENVIRONMENTastanza	375
ENVIRONMENTedarrayc	291
ENVIRONMENTedarrayl	291
ENVIRONMENTedarrayr	291
ENVIRONMENTedtabularc	291
ENVIRONMENTedtabularl	291
ENVIRONMENTedtabularr	291
ENVIRONMENTledgroup	66, 252, 375
ENVIRONMENTledgroup sized	252
PACKAGE(r)(e)ledmac	29
PACKAGEEledmac	11, 63, 87, 257, 312, 313, 370–372
PACKAGEEledpar	371, 372
PACKAGEEtoolbox	65
PACKAGEParallel	318
PACKAGEReledmac	316
PACKAGEamsgen	269
PACKAGEamsmath	269
PACKAGEbabel	62, 280, 375
PACKAGEbiblatex	59
PACKAGEbidi	67, 374
PACKAGEccaption	73
PACKAGEcolor	61
PACKAGEedmac	1, 5, 10–13, 63, 168, 174, 231, 264, 309, 318, 362
PACKAGEedstanza	1, 13, 263
PACKAGEeledmac	1, 9, 10, 13–15, 52, 117, 174, 254, 257, 273, 295, 306, 310, 312–314, 366, 368, 370
PACKAGEeledpar	73, 146, 293, 318, 365, 368–371

PACKAGEetex	374
PACKAGEtoolbox	78, 117, 206, 217, 225, 246, 293, 304
PACKAGEfootmisc	28, 61, 174, 318
PACKAGEglossaries	53, 375
PACKAGEhandout	369
PACKAGEhyperlink	213
PACKAGEhyperref	47, 111, 192, 231, 259, 298, 367–369, 376
PACKAGEifluatex	65
PACKAGEifxetex	65
PACKAGEmakeidx	51, 59, 66, 67, 254, 257, 313, 366–368, 370
PACKAGEindextols	261
PACKAGEindextool	313
PACKAGEindextools	51, 59, 66, 67, 73, 254, 257, 261, 313, 370, 375
PACKAGEinputenc	119
PACKAGEledarab	62
PACKAGEledmac	1, 10, 12, 13, 62, 78, 257, 309, 310, 313, 316
PACKAGEledpar	62
PACKAGEMemoir	66, 257, 313, 318, 369
PACKAGEMorewrites	60
PACKAGEMusixtex	368
PACKAGEperpage	375
PACKAGEpolyglossia	35, 62, 113, 144, 165, 375
PACKAGERagged2e	38, 65
PACKEREledmac	1, 2, 10, 11, 13–15, 17, 18, 21–23, 25–29, 31, 34, 37–39, 41, 43, 45–48, 51–53, 58–61, 63, 64, 79, 81, 87, 88, 91, 92, 94, 103, 110, 140, 147, 149, 154, 174, 199, 207, 211, 217, 225, 235, 238, 257, 273, 293, 294, 306, 313, 314, 316, 317, 373
PACKEREledpar	1, 4, 5, 8, 14, 17, 40, 47, 49, 58, 59, 62, 64, 73, 79, 90, 94, 112, 147, 150, 194, 195, 207, 212, 224, 226, 227, 253, 263, 374, 375
PACKAGESuffix	65
PACKGETabmac	1, 12, 13, 318
PACKAGEuninormalize	26
PACKAGExargs	25, 65
PACKAGExkeyval	64, 224
PACKAGExstring	65, 259

## 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

<code>\affixpstart@num</code>	1
<code>\affixside@note</code>	1
<code>\Afootnote</code>	23
<code>\afternoteX</code>	38
<code>\afterruleX</code>	39
<code>\ampersand</code>	1, 44
<code>\applabel</code>	1, 48
<code>\appref</code>	1, 48
<code>\apprefwithpage</code>	1, 48
<code>\arrangementX</code>	1, 30
<code>\arrangementX@normal</code>	1
<code>\arrangementX@threecol</code>	1
<code>\arrangementX@twocol</code>	1
<code>\at@every@pend</code>	1
<code>\AtEveryPend</code>	1, 17
<code>\AtEveryPstart</code>	1, 17
<code>\autopar</code>	1, 16

**B**

<code>\ballast</code>	60
<code>\ballast@count</code>	1
Beeton, Barbara Ann Neuhaus Friend	17
<code>\beforeeledchapter</code>	1
<code>\beforenotesX</code>	39
<code>\beginnumbering</code>	1, 15
<code>\Bendnote</code>	23
<code>\Bfootnote</code>	23
<code>\bhookgroupX</code>	39
<code>\bhooknoteX</code>	37
<code>\bodyfootmarkA</code>	28
<code>\boxfootnotenumbers</code>	1
Bredon, Simon	12
Breger, Herbert	12, 272
Brey, Gerhard	12
Busard, Hubert L. L.	12
<code>\bypage@false</code>	1
<code>\bypage@true</code>	1
<code>\bypstart@false</code>	1
<code>\bypstart@true</code>	1

**C**

<code>\c@addcolcount</code>	1
<code>\c@ballast</code>	1
<code>\c@firstlinenum</code>	1
<code>\c@firstsublinenum</code>	1
<code>\c@labidx</code>	1
<code>\c@linenumincrement</code>	1
<code>\c@sublinenumincrement</code>	1
<code>\Cendnote</code>	23
<code>\Cfootnote</code>	23

<code>\ch@ck@l@ck</code> .....	1
<code>\ch@cksub@l@ck</code> .....	1
<code>\chapter</code> .....	1
<code>\check@pb@in@verse</code> .....	1
Chester, Robert of .....	12
Claassens, Geert H. M. ....	12
<code>\colalignX</code> .....	37
Copernicus, Nicolaus .....	12
<code>\critext</code> .....	309
<code>\ctab</code> .....	1
<code>\ctabtext</code> .....	1

## D

Dekker, Dirk-Jan .....	61
<code>\Dendnote</code> .....	23
<code>\Dfootnote</code> .....	23
<code>\disable@familiarnotes</code> .....	1
<code>\disable@notes</code> .....	1
<code>\disable@sidenotes</code> .....	1
<code>\disablel@dtabfeet</code> .....	1
<code>\do@actions</code> .....	1
<code>\do@actions@fixedcode</code> .....	1
<code>\do@actions@next</code> .....	1
<code>\do@ballast</code> .....	1
<code>\do@feetX</code> .....	1
<code>\do@insidelinehook</code> .....	1
<code>\do@line</code> .....	1
<code>\do@linehook</code> .....	1
<code>\do@lockoff</code> .....	1
<code>\do@lockoffL</code> .....	1
<code>\do@lockon</code> .....	1
<code>\do@lockonL</code> .....	1
<code>\doedindexlabel</code> .....	1
<code>\doendnotes</code> .....	1, 23
<code>\doendnotesbysection</code> .....	1, 23
<code>\doinsidelinehook</code> .....	1, 21
<code>\dolinehook</code> .....	1, 21
<code>\dosplits</code> .....	1
Downes, Michael .....	60, 153, 155
<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, 286
<code>edarrayc</code> (environment) .....	53
<code>edarrayl</code> (environment) .....	53
<code>edarrayr</code> (environment) .....	53
<code>\edatleft</code> .....	1, 55

<code>\edatright</code>	1, 55
<code>\edbeforetab</code>	1, 56, 286
<code>\edfilldimen</code>	1
<code>\edfont@info</code>	1
<code>\edGLS</code>	1
<code>\edGls</code>	1
<code>\edgls</code>	1
<code>\edglsdisp</code>	1
<code>\edGLSpl</code>	1
<code>\edGlspl</code>	1
<code>\edglspl</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, 54
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 54
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1
<code>\EDTABtext</code>	1
<code>edtabularc (environment)</code>	53
<code>edtabularl (environment)</code>	53
<code>edtabularr (environment)</code>	53
<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> .....	53
<code>edarrayl</code> .....	53
<code>edarrayr</code> .....	53
<code>edtabularc</code> .....	53
<code>edtabularl</code> .....	53
<code>edtabularr</code> .....	53
<code>ledgroup</code> .....	45
<code>ledgroupsize</code> .....	45
<code>minipage</code> .....	45
Euclid .....	12
<code>\extensionchars</code> .....	1, 59

## F

<code>\f@x@l@cks</code> .....	1
Fairbairns, Robin .....	28
<code>\first@linenum@out@false</code> .....	1
<code>\first@linenum@out@true</code> .....	1
<code>\firstlinenum</code> .....	1, 18
<code>\firstseriesX@</code> .....	1
<code>\firstsublinenum</code> .....	1, 18
<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, 60
<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>\footplitskips</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>\hsizex</code> .....	40
<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>\if@eled@sectioning</code> .....	1
<code>\if@led@nofoot</code> .....	1
<code>\if@ledgroup</code> .....	1
<code>\if@lemmacommand@</code> .....	1
<code>\if@noeled@sec</code> .....	1
<code>\if@noneed@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@dhidenumber</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@ddodoreinextrafeet</code> .....	1
<code>\l@dedbeginmini</code> .....	1
<code>\l@dedendmini</code> .....	1
<code>\l@emptyd@ta</code> .....	1
<code>\l@dend@close</code> .....	1
<code>\l@dend@open</code> .....	1
<code>\l@dend@stuff</code> .....	1
<code>\l@dend@Xfalse</code> .....	1
<code>\l@dend@Xtrue</code> .....	1
<code>\l@denvbody</code> .....	1
<code>\l@dfambeginmini</code> .....	1
<code>\l@dfamendmini</code> .....	1
<code>\l@dfetbeginmini</code> .....	1

<code>\ldfeetendmini</code>	1
<code>\ldgetline@margin</code>	1
<code>\ldgetlock@disp</code>	1
<code>\ldgetref@num</code>	1
<code>\ldgetsidenote@margin</code>	1
<code>\ldgobblearg</code>	1
<code>\ldlabel@parse</code>	1
<code>\ldld@ta</code>	1
<code>\ldlp@rbox</code>	1
<code>\ldlsn@te</code>	1
<code>\ldlsnote</code>	1
<code>\ldmake@labels</code>	1
<code>\ldmodforedtext</code>	1
<code>\ldnullfills</code>	1
<code>\ldnumpstartsL</code>	1
<code>\ldoldold@footnotetext</code>	1
<code>\ldp@rsefootspec</code>	1
<code>\ldparsedendline</code>	1
<code>\ldparsedendpage</code>	1
<code>\ldparsedendsub</code>	1
<code>\ldparsedstartline</code>	1
<code>\ldparsedstartpage</code>	1
<code>\ldparsedstartsub</code>	1
<code>\ldpush@begins</code>	1
<code>\ldrd@ta</code>	1
<code>\ldref@undefined</code>	1
<code>\ldrestorefills</code>	1
<code>\ldrestoreforedtext</code>	1
<code>\ldrp@rbox</code>	1
<code>\ldrsn@te</code>	1
<code>\ldrsnote</code>	1
<code>\ldsetmaxcolwidth</code>	1
<code>\ldskipnumberfalse</code>	1
<code>\ldskipnumbertrue</code>	1
<code>\ldtabaddcols</code>	1
<code>\ldtabnoexpands</code>	1
<code>\ldunboxmpfoot</code>	1
<code>\ldunhbox@line</code>	1
<code>\ldzeropenalties</code>	1
<code>\label</code>	47
<code>\labelpstartfalse</code>	1
<code>\labelpstarttrue</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>ledgroup</code> (environment) .....	45
<code>ledgroupsize</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, 50
<code>\ledlsnotesep</code> .....	1, 50
<code>\ledlsnotewidth</code> .....	1, 50
<code>\lednopb</code> .....	1, 58
<code>\lednopbinversetrue</code> .....	58
<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, 58
<code>\ledrightnote</code> .....	1, 50
<code>\ledrlfill</code> .....	1
<code>\ledrsnotefontsetup</code> .....	1, 50
<code>\ledrsnotesep</code> .....	1, 50
<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, 19
<code>\leftltab</code> .....	1
<code>\leftnoteupfalse</code> .....	50
<code>\leftpstartnum</code> .....	1
<code>\leftrtab</code> .....	1
<code>Leibniz</code> .....	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, 24
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 18
<code>\linenumberstyle</code>	1, 21
<code>\linenumincrement</code>	1, 18
<code>\linenummargin</code>	1, 19
<code>\linenumr@p</code>	1
<code>\linenumrep</code>	1
<code>\linenumsep</code>	1, 19
<code>\linerrangesep@</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>\measurebody</code>	1
<code>\measuremcell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretcell</code>	1
<code>\measuretrow</code>	1
Middleton, Thomas	12, 86
<code>minipage</code> (environment)	45
Mittelbach, Frank	11, 12
<code>\morenoexpands</code>	1, 61
<code>\mpfnpos</code>	1, 29
<code>\mpnormalfootgroup</code>	1
<code>\mpnormalfootgroupX</code>	1
<code>\mpnormalvfootnote</code>	1
<code>\mpnormalvfootnoteX</code>	1
<code>\mppara@footgroupX</code>	1
<code>\mppara@vfootnoteX</code>	1
<code>\mpparafootgroup</code>	1



<code>\mpparavfootnote</code> .....	1
<code>\mpthreecolfootgroup</code> .....	1
<code>\mpthreecolfootgroupX</code> .....	1
<code>\mpthreecolfootsetup</code> .....	1
<code>\mpthreecolfootsetupX</code> .....	1
<code>\mptwocolfootgroup</code> .....	1
<code>\mptwocolfootgroupX</code> .....	1
<code>\mptwocolfootsetup</code> .....	1
<code>\mptwocolfootsetupX</code> .....	1
<code>\multfootsep</code> .....	1, 28
<code>\multiplefootnotemarker</code> .....	1

## N

<code>\n@num</code> .....	1
<code>\n@num@stanza</code> .....	1
<code>\newline</code> .....	1
<code>\newhookarg@specific</code> .....	1
<code>\newhookcommand@series</code> .....	1
<code>\newhookcommand@series@reload</code> .....	1
<code>\newhooktoggle@series</code> .....	1
<code>\newhooktoggle@series@reload</code> .....	1
<code>\newhooktoggle@specific</code> .....	1
<code>\newseries@</code> .....	1
<code>\newverse</code> .....	1
<code>\NEXT</code> .....	1
<code>\no@expands</code> .....	1
<code>\noeledsec</code> .....	58
<code>\nomk@</code> .....	1
<code>\normal@footnotemarkX</code> .....	1
<code>\normal@page@break</code> .....	1
<code>\normal@pars</code> .....	1
<code>\normalbfnoteX</code> .....	1
<code>\normalbodyfootmarkX</code> .....	1
<code>\normalfootfmt</code> .....	1
<code>\normalfootfmtX</code> .....	1
<code>\normalfootfootmarkX</code> .....	1
<code>\normalfootgroup</code> .....	1
<code>\normalfootgroupX</code> .....	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>optioninnnote</code>	375
<code>optioninnote</code>	375
<code>optionlinangesep</code>	224
<code>optionnocritical</code>	375
<code>optionnoend</code>	375
<code>optionnotenumber</code>	375

## 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, 17
<code>\pend</code>	1, 15
<code>Plato of Tivoli</code>	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, 15
<code>\pstarteref</code> .....	1
<code>\pstartnum</code> .....	1
<code>\pstartref</code> .....	46

## Q

<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, 89
<code>\resetprevpage@</code> .....	1
<code>\resetprevpage@num</code> .....	89

\restore@familiarnotes	1
\restore@notes	1
\restore@sidenotes	1
\resumenumbering	1, 17
\rightctab	1
\rightlinenum	1, 19
\rightltab	1
\rightnoteupfalse	50
\rightrtab	1
\rightstartnum	1
\rigidbalance	1
\rigidbalanceX	1
\rtab	1
\rtabtext	1

## S

Sacrobosco	12
\sameword	1, 26
\sameword@inedtext	1
Schöpf, Rainer	12
\section@num	1
\select@lemmafont	1
\select@lemmafont	1, 41
\SEref	1, 47
\SErefonlypage	48
\SErefwithpage	1, 48
\series	1
\seriesatbegin	1, 29
\seriesatend	1, 29
\set@line	1
\set@line@action	1
\setapprefprefixmore	48
\setapprefprefixsingle	48
\setcommand@series	1
\sethangingsymbol	1, 43
\setistwofollowinglines	1
\setl@dlp@rbox	1
\setl@drpr@box	1
\setline	1, 20
\setlinenum	1, 21
\setmcellcenter	1
\setmcellleft	1
\setmcellright	1
\setmrowcenter	1
\setmrowleft	1
\setmrowright	1
\setnoteswidthliketwocolumnsX@	1
\setnotesXpositionliketwocolumns@	1
\setprintendlines	1
\setprintlines	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>\settccllcenter</code>	1
<code>\settccllleft</code>	1
<code>\settccllright</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, 54
<code>\spreadtext</code>	1, 54
<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, 18
<code>\sublinenumr@p</code>	1
<code>\sublinenumrep</code>	1
<code>\sublineref</code>	1, 46
<code>\sublines@false</code>	1

\sublines@true	1
\sublock@disp	1
\sublockdisp	1
Sullivan, Wayne	12, 13, 60, 72, 77, 153, 154, 230, 263
\sza@penalty	1

## T

\tabHilfbox	1
\tabhilfbox	1
\theadcolcount	1
\theadtext	1
\theendpageline	1
\thefootnoteA	28
Theodosius	12
\thepageline	1
\thepstart	1, 17
\thestanza	1, 44
\thestartpageline	1
\this@line@list@version	1
\threecolfootfmt	1
\threecolfootfmtX	1
\threecolfootgroup	1
\threecolfootgroupX	1
\threecolfootsetup	1
\threecolfootsetupX	1
\threecolvfootnote	1
\threecolvfootnoteX	1
\twocolfootfmt	1
\twocolfootfmtX	1
\twocolfootgroup	1
\twocolfootgroupX	1
\twocolfootsetup	1
\twocolfootsetupX	1
\twocolvfootnote	1
\twocolvfootnoteX	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
--------------	----

\wrap@edcrossref .....	1
\wrapped@bodyfootmarkX .....	1
\wrapped@footfootmarkX .....	1
Wujastyk, Dominik .....	11

## X

\X@doreinfeet .....	1
\Xafterlemmaseparator .....	35
\Xafternote .....	38
\Xafternumber .....	33
\Xafterrule .....	39
\Xaftersymlinenum .....	33
\Xarrangement .....	1, 30
\Xarrangement@normal .....	1
\Xarrangement@paragraph .....	1
\Xarrangement@threecol .....	1
\Xarrangement@twocol .....	1
\Xbeforelemmaseparator .....	35
\Xbeforenotes .....	39
\Xbeforenumber .....	31, 33
\Xbeforesymlinenum .....	33
\Xbhookgroup .....	39
\Xbhooknote .....	37
\Xboxlinenum .....	34
\Xboxlinenumalign .....	34
\Xboxsymlinenum .....	34
\Xcolalign .....	37
\Xdo@feet .....	1
\xedindex .....	1
\xedlabel .....	1
\xedtext .....	1
\Xendafterrenumber .....	33
\Xendafterlemmaseparator .....	36
\Xendafternote .....	40
\Xendafterpagenumber .....	35
\Xendaftersymlinenum .....	33
\Xendahookinplaceofnumber .....	35
\Xendahooklinenumber .....	35
\Xendbeforelemmaseparator .....	36
\Xendbeforenumber .....	33
\Xendbeforepagenumber .....	35
\Xendbeforesymlinenum .....	33
\Xendbhookinplaceofnumber .....	35
\Xendbhooklinenumber .....	35
\Xendbhooknote .....	37
\Xendboxendlinenumalign .....	34
\Xendboxlinenum .....	34
\Xendboxlinenumalign .....	34
\Xendboxstartlinenumalign .....	34
\Xendboxsymlinenum .....	34

<code>\Xendhangindent</code>	37
<code>\Xendinplaceoflemmaseparator</code>	36
<code>\Xendinplaceofnumber</code>	34
<code>\Xendlemmadisablefontselection</code>	36
<code>\Xendlemmafont</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>\Xendsymlinenum</code>	31
<code>\Xendtwolines</code>	32
<code>\Xendtwolinesbutnotmore</code>	32
<code>\xflagref</code>	<u>1</u>
<code>\Xhangindent</code>	37
<code>\Xhsize</code>	40
<code>\Xhsizethreecol</code>	38
<code>\Xhsizetwocol</code>	38
<code>\Xinplaceoflemmaseparator</code>	35
<code>\Xinplaceofnumber</code>	33
<code>\Xinsertparafootsep</code>	<u>1</u>
<code>\Xledsetnormalparstuff</code>	<u>1</u>
<code>\xleft@appenditem</code>	<u>1</u>
<code>\Xlemmadisablefontselection</code>	36
<code>\Xlemmafont</code>	36
<code>\Xlemmaseparator</code>	35
<code>\Xlinerrangeseparator</code>	31
<code>\xlineref</code>	<u>1</u> , 46
<code>\Xmaxhnotes</code>	39
<code>\Xmorethantwolines</code>	31
<code>\Xnolemmaseparator</code>	<u>1</u> , 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>	32
<code>\xpageref</code>	<u>1</u> , 46
<code>\Xparafootsep</code>	38
<code>\Xparindent</code>	37



\Xpstart .....	32
\Xpstarteverytime .....	32
\xpstartref .....	<u>1</u> , 46
\Xragged .....	38
\xright@appenditem .....	<u>1</u>
\Xrigidbalance .....	<u>1</u>
\Xstanza .....	33
\Xstanzaseparator .....	33
\xsublineref .....	<u>1</u> , 46
\Xsublinesep .....	33
\Xsymlinenum .....	31
\Xtwolines .....	31
\Xtwolinesonlyinsamepage .....	32
\Xtxtbeforenotes .....	39
\Xunvxh .....	<u>1</u>
\xxref .....	<u>1</u> , 47

**Z**

\zz@@@ .....	<u>1</u>
--------------	----------

## 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	110
\reledmac@error: Added \eledmac@error and replaced error messages	67
v0.2.1.	
\@lab: Removed page setting from \@lab	233
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	228
\doxtrafeet: Renamed \doxtrafeet to \l@ddoxtrafeet	225
\edlabel: Tweaked \edlabel to get correct page numbers	231
\l@ddodoreinxtrafeet: Renamed \dodoreinxtrafeet to \l@ddodoreinxtrafeet	226
\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	110
\zz@@@: Minor change to \zz@@@	230
v0.2.2.	
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts package	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	233
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	92
General: Includes edstanza and more	1
\ledlinenum: Added \linenumr@p and \sublinenum@repto \leftlinenum and \rightlinenum	84
\linenumberlist: Added \linenumberlist mechanism	72
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	204
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	173
\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 critical footnotes	251
General: Added final/draft options	64
Added ledgroup environment	252
Added ledgroupsize environment	252
Added minipage, etc., support	1

\edtext: Added \showlemma to \edtext	111
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	249
\mpnormalfootgroup: Added \mpnormalfootgroup	150
\mpnormalvfootnote: Added \mpnormalvfootnote	148
\showlemma: Added \showlemma	72
\Xarrangement@normal: Added minpage footnote setup to \footnormal	147
v0.4.1.	
General: Added code for changing \docclearpage	228
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	175
\edindex: Let eledmac take advantage of memoir's indexing	257
\print@Xnotes: Added \@opXfeet	226
\Xdo@feet: Changed \Xdo@feet code for easier extensions	225
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	175
\@xympar: Eliminated \marginpar disturbance	243
General: Added left and right side notes	243
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
General: Added moveable side note	243
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place	252
\affixline@num: Changed \affixline@num to cater for sidenotes	135
\l@dgetsidenote@margin: Added \sidenotemargin and \sidenote@margin	243
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	94
\@nl@reg: Added \fix@page to \@nl	92
Extended \@nl to include the page number	92
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
\fix@page: Added \last@page@num and \fix@page	94
\get@thisfootnote: Changed \l@dbfnote and \vl@dbfnote as originals could give incorrect markers in the footnotes	176
\new@line: Extended \new@line to output page numbers	104
v0.7.0.	
\@nl@reg: Added \@nl@reg	92
\@ref@reg: Added \@ref@reg	101
General: eledmac having been available for 2 years, deleted the commented out original edmac texts	1
Maïeul Rouquette new maintainer	1
Made macros of all messages	67
Replaced all \interAfootnotelinepenalty, etc., by just \interfootnotelinepenalty	1
Tidying up for eledpar and ledarab packages	1
\affixline@num: Added skipnumering to \affixline@num	135
\do@actions@fixedcode: Added \do@actions@fixedcode	134

\do@actions@next: Added number skipping to \do@actions	133
\do@insidelinehook: Added \do@linehook for use in \do@line	131
\endnumbering: Changed \endnumbering for eldpar	76
\f@x@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \f@x@l@cks	137
\footsplitskips: Added \footsplitskips for use in many footnote styles	146
\get@linelistfile: Added \get@linelistfile	91
\initnumbering@reg: Added \initnumbering@reg	74
\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot containing some common code	251
\l@dcnotetext@r: Added \l@emptyd@ta	131
\l@dgetline@margin: Added \l@dgetline@margin	80
\l@dgetlock@disp: Added \l@dgetlock@disp	83
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	243
\l@dnumpstartsL: Added \l@dnumpstartsL, \ifl@dpairing and \ifpst@rted for/from eldpar	73
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	131
\l@dzeropenalties: Added \l@dzeropenalties	126
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	84
\line@list@stuff: Deleted \page@start from \line@list@stuff	104
\list@clearing@reg: Added \list@clearing@reg	91
\n@num: Added \n@num	99
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	181
\resumenumbering: Changed \resumenumbering for eldpar	77
\setprintendlines: Added \setprintendlines for use by \printendlines	203
\setprintlines: Added \setprintlines for use by \printlines	169
\skipnumbering: Added \skipnumbering and supports	107
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	82
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	84
Using \sublinenumrep instead of \sublinenumr@p	84
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	183
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 \edtext ; \critex ; \lemma fixed: we can now use non-switching commands	1
v0.9.0.	
General: No more ledpatch. All patches are now in the main file.	1
v0.9.1.	
General: Fix some bugs linked to integrating ledpatch on the main file.	1
v0.10.0.	
General: Corrections to \section 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 \hangingsymbol to define the character.	1
v0.12.0.	
General: For compatibility with eldpar, possibility to use \autopar on the right side.	1
Possibility to number \pstart.	17
Possibility to number the pstart with the commands \numberpstarttrue.	1

<code>\l@dnumpstartsL</code> : Added <code>\ifledRcol</code> and <code>\ifnumberingR</code> for/from <code>eledpar</code> . . . .	73
v0.12.1.	
General: Don't number <code>\pstarts</code> of stanza. . . . .	1
The numbering of <code>\pstarts</code> restarts on each <code>\beginnumbering</code> . . . . .	1
v0.13.0.	
General: New <code>stanzaindentsrepetition</code> counter to repeat stanza indents every $n$ verses. . . .	42
New <code>stanzaindentsrepetition</code> counter: to repeat stanza indents every $n$ verses. . . . .	1
<code>\managestanza@modulo</code> : New <code>stanzaindentsrepetition</code> counter to repeat stanza indents every $n$ verses. . . . .	265
v0.13.1.	
General: <code>\thepstartL</code> and <code>\thepstartR</code> use now <code>\bfseries</code> and not <code>\bf</code> , which is deprecated and makes conflicts with <code>memoir</code> class. . . . .	1
v0.14.0.	
General: Tweaked <code>\edlabel</code> to get correct line number if the command is first element of a paragraph. . . . .	1
<code>\edlabel</code> : Tweaked <code>\edlabel</code> to get correct line number if the command is first element of a paragraph. . . . .	231
v0.15.0.	
General: Line numbering can be reset at each <code>pstart</code> . . . . .	79
Possibility to print <code>\pstart</code> number inside. . . . .	17
<code>\affixline@num</code> : Line numbering can be disabled. . . . .	135
<code>\ifinserthangingsymbol</code> : New management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . . . .	263
<code>\printlines</code> : Line numbering can be reset at each <code>pstart</code> . . . . .	172
v0.17.0.	
<code>\ifinserthangingsymbol</code> : New new management of <code>hangingsymbol</code> insertion, preventing undesirable insertions. . . . .	263
v1.0.0.	
General: <code>\lemma</code> can contain commands. . . . .	24
Debug in lineation command . . . . .	19
New generic commands to customize footnote display. . . . .	29, 217
Options <code>nonum</code> and <code>nosep</code> in <code>\Xfootnote</code> . . . . .	23
Options of <code>\Xfootnotes</code> . . . . .	143
Possibility to have commands in sidenotes. . . . .	50
Some compatibility break with <code>eledmac</code> . Change of name: <code>eledmac</code> . . . . .	1
<code>\morenoexpands</code> : Change to be compatible with new features . . . . .	110
v1.0.1.	
General: Correction on <code>\Xnumberonlyfirstinline</code> with lineation by <code>pstart</code> or by page. . . .	31
v1.1.0.	
General: Add <code>\labelpstarttrue</code> . . . . .	17
Add <code>\Xnumberonlyfirstintwolines</code> . . . . .	31
Add <code>\Xpstart</code> and <code>\Xonlypstart</code> . . . . .	32
New hook to add arbitrary code at the beginning of the notes . . . . .	37
New options for block of notes. . . . .	39
New package option: <code>parapparatus</code> . . . . .	1
New tools to change order of series . . . . .	216
Sectioning commands. . . . .	57
<code>\preXnotes</code> : New skip <code>\preXnotes@</code> . . . . .	196
<code>\settoggle@series</code> : <code>\settoggle@series</code> switch the global value of the toggle, not only the local value. . . . .	217

v1.2.0.	
<code>\endquote</code> :	Compatibility of <code>\ledchapter</code> with the <i>memoir</i> class. . . . . 291
<code>\preXnotes</code> :	Debug in familiar footnotes (bug introduced by v1.1). . . . . 196
v1.3.0.	
<code>\endquote</code> :	<i>Quotation</i> and quote environment inside numbered sections. . . . . 291
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). 111
<code>\ledsetnormalparstuffX</code> :	Direction of footnotes with polyglossia. . . . . 193
<code>\newseries@</code> :	Remembers the language of the lemma, in order to create a correct direction for the footnote separator. . . . . 208
<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. . . . . 248
<code>\endquote</code> :	New option <i>noquotation</i> . . . . . 291
<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> . . . . . 232
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> . . . . . 291
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. . . . . 181
v1.5.0.	
General:	Correct indexing when the call is made in critical notes. . . . . 254
<code>\do@insidelinehook</code> :	Added <code>\do@insidelinehook</code> for use in <code>\do@oline</code> . . . . . 131
<code>\edindex</code> :	Compatibility with <code>imakeidx</code> package, and possibility to use multiple index with <code>\edindex</code> . . . . . 257
v1.5.1.	
<code>\managestanza@modulo</code> :	Correct <code>stanzaindentsrepetition</code> counter . . . . . 265
<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. . . . . 267
v1.6.1.	
General:	Corrects a false hanging verse when a verse is exactly the length of a line. . . . . 1

\AtEveryPstart: Spurious space in \pstart. . . . .	123
\ifinserthangingsymbol: Hang verse is now not automatically flush right. . . . .	263
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill in-side. . . . .	128
\pend: Spurious space in \pend. . . . .	125
v1.7.0.	
General: New features for managing page breaks. . . . .	58
v1.8.0.	
General: Compatibility with parledgroup option ofeledpar package. . . . .	1
If imakeidx and hyperref are loaded, adds hyperref in the index. . . . .	254
\endquote: Correction of sectioning commands in parallel texts. . . . .	291
\get@index@command: Debug \get@index@command and compatibility with hyperref package. . . . .	256
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work. . . . .	220
\prevpage@num: Correct \parafootsep when using with ledgroup. . . . .	158
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined). . . . .	198
v1.8.2.	
General: Debug compatibility problem with hebrew option of babel package. . . . .	1
v1.8.3.	
General: Fixes spurious spaces added by v1.7.0. . . . .	1
v1.8.5.	
General: Debug indexing in right column, witheledpar. . . . .	254
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes. . . . .	225
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / led-group. . . . .	249
v1.10.0.	
General: Add \pstartref and \xpstartref to refer to a pstart number (extension of \edlabel). . . . .	1
\endquote: Correction of sectioning commands in parallel texts. . . . .	291
v1.10.1.	
General: Compatibility with cleveref. . . . .	1
v1.10.2.	
General: Compatibility of stanza with v1.8a of babel-greek. . . . .	1
v1.10.3.	
General: Debug of cross-referencing. . . . .	1
v1.10.4.	
General: Debug of critical notes in edtabular environment. . . . .	1
v1.10.5.	
General: Debug of \pausenumbering. . . . .	1
Debug of \xxref. . . . .	1
v1.10.6.	
General: Debug of interaction between \autopar and \pausenumbering. . . . .	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 \musixtex, \@l becomes \@l. Consequently, \@l@reg becomes \@nl@reg. . . . .	92
General: Add \ledinnernote and \ledouternote commands. . . . .	50
Add \Xendparagraph and related settings. . . . .	40
Add hyperlink to crossref (needs hyperref package). . . . .	45
Compatibility with musixtex. . . . .	1
Debug eledmac sectioning command after using \resumenumbering. . . . .	1
Ensure that imakeidx is loaded <i>before</i> eledmac . . . . .	254
New hooks: \Xafterrule and \afterruleX . . . . .	39
New options for ragged-paragraph notes . . . . .	38
New sectioning commands. . . . .	57
Optional arguments for \pstart and \pend. . . . .	17
\AtEveryPstart: New optional argument for \pstart, to execute code before it. . .	123
\edindex: Use correctly default index when imakeidx is loaded. . . . .	257
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands. . . . .	291
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar. 74	
\l@dcsnote: \l@dlsnote, \l@drsnote and \l@dcsnote defined only one time, in eledmac, including needs for eledpar case. . . . .	245
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar. . . . .	243
\l@dunpstartsL: Add \ifledRcol@ for eledpar . . . . .	73
\l@dunhbox@line: \do@line is split in more little commands. . . . .	129
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX. . . . .	220
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph. . . . .	220
\pend: New optional argument for \pend, to execute code after it. . . . .	125
\stanza: &can have an optional argument: content to be printed after. . . . .	267
\Stanza can have an optional argument: content to be printed before. . . . .	267
Add \newverse macro, \falseverse deprecated. . . . .	267

## v1.12.1.

\wrap@edcrossref: Fix spurious spaces. . . . .	235
--	-----

## v1.12.2.

\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0)	128
--	-----

## 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 \eledxxx with some paper size . . . . .	1
Debug \ledinnernote and \ledouternote commands in the top of pages. . . . .	50
Debug left and right notes (bugs added by 1.12.0) . . . . .	1
Underline lemma in \eledxxx when using draft mode. . . . .	1
\flag@end: \flag@start and \flag@end are now defined only one time for eledmac and eledpar . . . . .	105
\flag@start send a error message when a \edtext is done without insert (note)	105
\reledmac@error: Replaced error messages . . . . .	67

## v1.12.4.

General: Debug spurious page breaks before \chapter (bug added in 1.12.0) . . . . .	1
---	---



v1.12.5.	
\@edindex@hyperref: Debug \edindex when hyperref is not loaded	259
\@ssect: Debug \eledchapter in parallel with memoir	295
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	131
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	76
\l@dgobblearg: \l@dgobblearg becomes \l@dgobbeloptarg	275
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	275
\resumenumbering: Debug \resumenumbering	77
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	235
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	105
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	40
Added widthliketwocolumns option	64
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	220
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	189
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	217
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@dzero penalties 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.	198
Delete previous override of \l@d@wrindexhyp at the beginning of a document when hyperref is not loaded.	261
Move gobbling command	72
Provide \@gobblefour	72
\edindex: Let eledmac take advantage of imakeidx even when memoir class is used	257
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref package.	298
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	295
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 . . . . .	25
\arrangementX@threecol: Correct bug with paragraphed familiar footnotes setting. . . . .	188
\endsub: Restore subline feature (disabled by mistake in v1.8.0). . . . .	105
\if@lemmacommand@: New boolean \iflemmacommand@. . . . .	116
v1.15.1.	
\line@list@stuff: 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: \edtext is now defined only in eledmac, not in eledpar. Debug wrong num- bering when using \sameword + eledpar + \tag command. . . . .	111
Compatibility of standard footnotes with some biblatex styles. . . . .	1
New \stanzaindent command. . . . .	1
v1.16.1.	
\xlineref: \lineref is not defined if defined by some other package, like lineno. Eledmac provides \edlineref instead. . . . .	235
v1.17.0.	
\edtext: Error message when calling \edtext outside of a numbered paragraph. . . .	111
v1.18.0.	
\@edindex@hyperref: Fix spurious space with \edindex when using imakeidx/indextools + hyperref. . . . .	259
General: Add \Xpstarteverytime . . . . .	32
Compatibility with Lua <sup>®</sup> TeX RTL languages. . . . .	1
Debug \Xonlypstart when using \Xnumberonlyfirstinline and the current line number differs from the previous. . . . .	32
\edlabel: \edlabel is now defined only one time for both eledmac and eledpar . . .	231
\l@d@section: Option parapparatus works for endnotes. . . . .	199
\l@dnumstartsl: Add \ifl@dprintingpages and \dprintingcolumns for eledpar . . . . .	73
\print@line: Compatibility with Lua <sup>®</sup> TeX RTL languages. . . . .	129
\printlinefootnote: Code refactoring in \printlinefootnote: the printing of the numbers are factorized in \printlinefootnotearea . . . . .	165
\printpstart: Debug \Xpstart with parallel pages and columns (eledpar) . . . . .	165
v1.19.0.	
General: \Xmaxhnotes and \maxhnotesX work now for both two-columns and three- columns setting. . . . .	1
Compatibility with eledpar v1.13.0. . . . .	1
\footsplitskips: \footsplitskips doesn't set \floatingpenalty to \@MM when processing parallel pages. . . . .	146
\xxref: \xxref works also with right side numbers, when \@Rlineflag is not empty. .	237
v1.19.1.	
General: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages, that is in eledpar. . . . .	1
v1.20.0.	
General: Add \Xendboxlinenum . . . . .	34
Add \Xtwolines and \Xmorethantwolines hooks . . . . .	31
Add series option. . . . .	1
Correct \Xinplaceofnumber hook. . . . .	1
Explicit error message when calling \Xfootnote outside of \edtext. . . . .	1
Fix bug with line number typesetting direction when using \eledsection and similar commands for RTL texts with Lua <sup>®</sup> 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@Xmorethantwoline and $\backslash$ ifl@d@Xmorethantwoline to $\backslash$ printlines	173
$\backslash$ stanza: & and $\backslash$ & can be preceded by spaces.	267
$\backslash$ xxref: Debug $\backslash$ xxref when not loading eledpar (fix bug added in 1.19.0).	237
v1.21.0.	
$\backslash$ @edindex@hyperref: Look at the hyperindex option of hyperref before inserting hyperref	259
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	216
Add $\backslash$ applabel and related	48
Add $\backslash$ beforenotesX and $\backslash$ Xbeforenotes features for notes set in two and three column.	1
Add $\backslash$ hidenumbering	21
Add $\backslash$ Xcolalign and $\backslash$ colalignX hooks	37
Add $\backslash$ Xendtwoline, $\backslash$ Xendmorethantwoline, $\backslash$ Xendtwolinebutnotmore and $\backslash$ Xendtwolineonlyinsamepage.	32
Add $\backslash$ Xparindent and $\backslash$ hangindentX	37
Add $\backslash$ Xtwolinebutnotmore and $\backslash$ Xtwolineonlyinsamepage.	1
Add nocritical, noend, nofamiliar and noledgroup options.	1
Add noledsec 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. oldprintnnumspace 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.	199
$\backslash$ ledinnotemark: Add $\backslash$ ledinnotemark.	257
$\backslash$ ledsetnormalparstuffX: $\backslash$ ledsetnormalparstuff is deprecated and becomes $\backslash$ ledsetnormalparstuffX and $\backslash$ Xledsetnormalparstuff.	193
$\backslash$ n@num: $\backslash$ n@num@ref deleted	99
$\backslash$ n@num defined only one time for both Eledmac and Eledpar	99
$\backslash$ newhookcommand@series: $\backslash$ newhookcommand@series can take an optional argument.	219
$\backslash$ newhooktoggle@series: $\backslash$ newhooktoggle@series can take an optional argument.	220

\print@footnoteXrule: Code refactoring: the spaces after the footnote rules are directly managed in \print@Xfootnoterule and \print@footnoteXrule	195
\seriesatend: Fix spurious space in \seriesatend	216
\skipnumbering: \skipnumbering defined only one time for both Eledmac and Eledpar.	107
Correct \skipnumbering for stanza.	107
Delete \skipnumbering@reg.	107
v1.22.0.	
General: Add \doendnotesbysection command.	23
Add option for lemma separator inside endnotes	36
Adds hyperlink for references to notes in indices.	1
Fix conflict between noend package option and edtabularx environments	1
Provides support for xindy.	1
Standardize endnotes handbook.	23
When using hyperref package, internal links in index or with \edlineref are now targeted to the top and not longer to the bottom of the lines they refer to.	1
\ledinnote: \ledinnote takes a first optional argument, which is the label for hyperlinks.	257
v1.22.1.	
General: Fix bug (added on v1.22.0) with \Xinplaceofnumber hook.	1
\prevpage@num: Correct double symbol when using both \parafootsep and \Xsymlinenum.	158
v1.23.0.	
\@edtext@level: The boolean \if@edtext@ becomes the counter \edtext@level.	111
General: Add \Xboxlinenumalign and \Xendboxlinenumalign.	34
Add \Xboxstartlinenum, \Xendboxstartlinenum, \Xboxendlinenum, \Xendboxendlinenum.	34
Allow use of \sameword with inputenc managing of UTF-8.	1
Compatibility between nofamiliar/nocriticals option and minipage/ledgroup.	1
Error message when using \beginnumbering... \endnumbering without \pstart.	1
Fix bug with \sameword when the lemma overlaps multiple line.	25
Fix bug with \sameword when the same lemma is used for multiple notes or for nested \edtexts.	25
Fix bug with \skipnumbering called immediately after a \pstart.	1
Fix error of \iftrue not closed.	1
Fix spurious space with \skipnumbering (bug added on v1.21.0).	1
New tools to ensure the line-list file uses the right version of commands when upgrading the eledmac version.	1
Optional argument of \sameword can be a comma-separated list of \edtext depth.	25
\lemma: Fix spurious space after \lemma command	115
\newseries@: Prevent spurious spaces when \Afootnote and similar commands are followed by spaces (bug added on 1.0.0).	208
\sameword: In order to allow use of \sameword with inputenc, we detokenize its mandatory argument before using it in control sequence names.	119
\SErefwithpage: Debug \Xendtwolines, \Xendmoreethantwolines, \Xendtwolinesbutnotmore and \Xendtwolinesonlyinsamepage when using \apprefwithpage.	240
v1.23.1.	
General: Fix bug with \lemma 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. ....	251
<code>\@xympar</code> : Patching <code>\@xympar</code> instead of redefining it ....	243
General: <code>\@makecol</code> , <code>\@reinserts</code> and <code>\@doclearpage</code> are patched instead of begin redefined ....	228
<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> . ....	228
Add <code>\Xendinplaceofnumber</code> hook. ....	1
Add <code>\Xendnonumber</code> hook. ....	1
Add <code>nonum</code> 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 ....	175
Suppress indexing command specific to <code>memoir</code> . ....	257
<code>\endminipage</code> : Patch <code>\endminipage</code> instead of redefining it. ....	251
<code>\initnumbering@quote</code> : <code>\initnumbering@sectcmd</code> becomes <code>\initnumbering@quote</code> ....	291
<code>\l@advance@parledgroup@beforenormalnotes</code> : Some conde of <code>\l@dumboxmpfoot</code> moved to <code>\l@advance@parledegroupp@beforenormalnotes</code> ....	251
<code>\newseries@</code> : One endnotes file by series. ....	213
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>\advanceline</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>\hooknoteX</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 <code>tabular</code> 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. . . . .	193
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 $\text{\LaTeX}$ . . . . .	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>\Xendbeforemsymmlinenumber</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>linrangesep</code> for critical footnotes and endnotes. . . . .	31
<code>\footnoteoptions@</code> : First argument of <code>\footnoteoption@</code> is now mandatory, not optional. . . . .	143

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 $\LaTeX$ <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 $\LaTeX$ <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 <code>\Xfootnote</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>\Xhookgroup</code> and <code>\hookgroupX</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	231
<code>\l@d@section</code> : <code>\endnotes</code> take six arguments.	199
<code>\printlines</code> : <code>\printlines</code> takes an eighth argument: the line flag	172
<code>\Serefwithpage</code> : Debug <code>\setapprefprefixsingle</code>	239
<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.	235
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 to <code>\Xrigidbalance</code> and <code>\rigidbalanceX</code> .	158
General: Add dash as default page range separator for <code>\SEonlypage</code>	1
Debug <code>\SErefonlypage</code> when referring to only one page.	1
Delete parenthesis after <code>\SErefonlypage</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>innnote</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 related to page and linenummer with the <code>glossaries</code> package	1
New hooks: <code>\Xlemmafont</code> and <code>\Xendlemmafont</code>	36
New setting commands: <code>\setSErefonlypageprefixsingle</code> and <code>\setSErefonlypageprefixmore</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@d@section: \endnotes</code> take seven arguments. . . . .	199