

# 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 . . . . .	10
1.2.1 edmac . . . . .	10
1.2.2 ledmac . . . . .	12
1.2.3 eledmac . . . . .	12

---

\*This file (**reledmac.dtx**) has version number v2.4.0, last revised 2015/10/19.

<sup>†</sup>maieul at maieul dot net



1.2.4 <code>reledmac</code> . . . . .	12
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>13</b>
3.1 Specific features . . . . .	13
3.2 Optimizing package performance . . . . .	14
<b>4 Text lines and paragraphs numbering</b>	<b>14</b>
4.1 Text lines numbering . . . . .	14
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> . . . . .	16
4.2.4 Content before every <code>\pstart</code> and after every <code>\pend</code> . . . . .	16
4.2.5 Numbering paragraphs ( <code>\pstart</code> ) . . . . .	16
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 . . . . .	18
4.3.4 Setting line number margin . . . . .	18
4.3.5 Other settings . . . . .	19
4.4 Changing the line numbers . . . . .	19
4.4.1 Sublineation . . . . .	19
4.4.2 Locking lineation . . . . .	19
4.4.3 Setting and changing line number . . . . .	20
4.4.4 Line number style . . . . .	20
4.4.5 Skipping and hiding number . . . . .	20
4.4.6 Execute code at each line . . . . .	20
<b>5 Apparatus commands</b>	<b>21</b>
5.1 Terminology . . . . .	21
5.2 Critical notes . . . . .	21
5.2.1 The lemma . . . . .	21
5.2.2 Footnotes . . . . .	22
5.2.3 Endnotes . . . . .	22
5.2.4 Paragraph in critical apparatus . . . . .	23
5.2.5 Change lemma and line number . . . . .	24
5.2.6 Changing the names of commands for critical apparatus . . . . .	24
5.3 Disambiguation of identical words in the apparatus . . . . .	25
5.3.1 Basic use . . . . .	25
5.3.2 Notes about input encoding with UTF-8 processor . . . . .	25
5.3.3 Use with <code>\lemma</code> command . . . . .	26



5.3.4 Customizing . . . . .	27
5.4 Familiar notes . . . . .	27
5.4.1 Basic use . . . . .	27
5.4.2 Customizing mark . . . . .	27
5.4.3 Separator for multiple footnotes . . . . .	28
5.5 Changing series . . . . .	28
5.5.1 Create a new series . . . . .	28
5.5.2 Delete series . . . . .	28
5.5.3 Series order . . . . .	28
5.6 Position of critical and familiar footnotes . . . . .	28
<b>6 Critical apparatus appearance</b> . . . . .	<b>29</b>
6.1 Notes arrangement in a series . . . . .	29
6.2 Control line number printing . . . . .	30
6.2.1 Print line number only at first time . . . . .	30
6.2.2 Separator for line range . . . . .	30
6.2.3 Abbreviate line range . . . . .	31
6.2.4 Disable line number . . . . .	31
6.2.5 Printing pstart number . . . . .	31
6.2.6 Printing stanza number . . . . .	32
6.2.7 Space around number . . . . .	32
6.2.8 Space around line symbol . . . . .	32
6.2.9 Space in place of number . . . . .	33
6.2.10 Boxing line number and line symbol . . . . .	33
6.3 Arbitrary code around line number . . . . .	34
6.4 Separator between the lemma and the note . . . . .	34
6.4.1 For footnotes . . . . .	34
6.4.2 For endnotes . . . . .	34
6.5 Font style . . . . .	35
6.5.1 For line number . . . . .	35
6.5.2 For the lemma . . . . .	35
6.5.3 For all notes . . . . .	35
6.6 Indent of notes content . . . . .	35
6.7 Arbitrary code at the beginning of notes . . . . .	36
6.8 Options for footnotes in columns . . . . .	36
6.8.1 Alignment . . . . .	36
6.8.2 Size of the columns . . . . .	37
6.9 Options for paragraphed footnotes . . . . .	37
6.9.1 Mark separation of notes . . . . .	37
6.9.2 Ragged text . . . . .	37
6.10 Options for block of notes . . . . .	37
6.10.1 Text before notes . . . . .	37
6.10.2 Spacing . . . . .	37
6.10.3 Rule . . . . .	38
6.10.4 Maximum height . . . . .	38
6.11 Footnotes and the reledpar columns . . . . .	38



6.12 Endnotes in one paragraph . . . . .	39
<b>7 Fonts</b>	<b>39</b>
<b>8 Verse</b>	<b>40</b>
8.1 Basic . . . . .	40
8.2 Define stanza indents . . . . .	40
8.3 Repeating stanza indents . . . . .	40
8.4 Manual stanza indent . . . . .	41
8.5 Stanza breaking . . . . .	41
8.6 Hanging symbol . . . . .	42
8.7 Long verse and page break . . . . .	42
8.8 Content before/after verses . . . . .	42
8.9 Numbering stanza . . . . .	42
8.10 Various tools . . . . .	43
8.11 Notes on empty lines . . . . .	43
<b>9 Grouping</b>	<b>43</b>
<b>10 Cross referencing</b>	<b>44</b>
10.1 Basic use . . . . .	44
10.2 Refer to a critical notes . . . . .	44
10.3 Cross-referencing which return a number in any case . . . . .	45
10.3.1 Cross-referencing in order to define line number of a critical note . . . . .	45
10.4 Not automatic cross-referencing . . . . .	45
10.5 Normal L <sup>A</sup> T <sub>E</sub> X cross-referencing . . . . .	46
10.6 References to lines commented in the apparatus . . . . .	46
<b>11 Side notes</b>	<b>47</b>
11.1 Basics . . . . .	47
11.2 Setting . . . . .	47
11.2.1 Width . . . . .	47
11.2.2 Vertical position . . . . .	47
11.2.3 Distance to the main text . . . . .	47
11.2.4 Separator between notes . . . . .	48
<b>12 Indexing</b>	<b>48</b>
12.1 Basics . . . . .	48
12.2 Separator between page and line numbers . . . . .	48
12.3 Using xindy . . . . .	49
12.4 Advanced setting . . . . .	49
<b>13 Tabular material</b>	<b>50</b>



<b>14 Sectioning commands</b>	<b>53</b>
14.1 Sectioning commands without line numbers or critical notes . . . . .	53
14.2 Sectioning commands with line numbering and critical notes . . . . .	53
14.3 Optimization . . . . .	54
<b>15 Quotation environments</b>	<b>54</b>
<b>16 Page breaks</b>	<b>54</b>
16.1 Control page breaking . . . . .	54
16.2 Prevent page break in a long verses . . . . .	55
<b>17 Miscellaneous</b>	<b>55</b>
17.1 Known and suspected limitations . . . . .	56
17.2 ‘No room for a new’ . . . . .	56
17.3 Marginal notes . . . . .	56
17.4 Paragraph shape . . . . .	56
17.5 Paragraphed footnotes . . . . .	57
17.6 Use with other packages . . . . .	57
17.7 Parallel typesetting . . . . .	58
<b>I Implementation overview</b>	<b>59</b>
<b>II Preliminaries</b>	<b>59</b>
II.1 Links with original edmac . . . . .	59
II.2 Package declaration . . . . .	59
II.3 Package options . . . . .	60
II.4 Loading packages . . . . .	61
II.5 Compatibility with LuaTeX . . . . .	62
II.6 Boolean flags . . . . .	62
II.7 Messages . . . . .	63
II.8 Gobbling . . . . .	68
II.9 Miscellaneous commands . . . . .	68
II.10 Prepare reledpar . . . . .	69
<b>III Sectioning commands</b>	<b>69</b>
<b>IV List macros</b>	<b>73</b>
<b>V Line counting</b>	<b>74</b>
V.1 Choosing the system of lineation . . . . .	74
V.2 Line number margin . . . . .	76
V.3 Line number initialization and increment . . . . .	77
V.4 Line number locking . . . . .	78
V.5 Line number style . . . . .	79
V.6 Line number printing . . . . .	80
V.7 Line number counters and lists . . . . .	80
V.8 Line number locking counter . . . . .	82



V.9 Line number associated to lemma . . . . .	82
V.10 Reading the line-list file . . . . .	85
V.11 Commands within the line-list file . . . . .	87
V.12 Writing to the line-list file . . . . .	98
<b>VI Marking text for notes</b>	<b>103</b>
VI.1 <code>\edtext</code> itself . . . . .	104
VI.2 Substitute lemma . . . . .	110
VI.3 Substitute line numbers . . . . .	111
VI.4 Lemma disambiguation . . . . .	112
<b>VII Paragraph decomposition and reassembly</b>	<b>118</b>
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	118
VII.2 Processing one line . . . . .	123
VII.2.1 General process . . . . .	123
VII.2.2 Process for “normal” line . . . . .	124
VII.2.3 Process for line containing <code>\eledsection</code> command . . . . .	125
VII.2.4 Hooks . . . . .	126
VII.2.5 Sidenotes and marginal line number initialization . . . . .	126
<b>VIII Line and page number computation</b>	<b>127</b>
<b>IX Line number printing</b>	<b>130</b>
<b>X Pstart number printing in side</b>	<b>134</b>
<b>XI Restoring footnotes and penalties</b>	<b>135</b>
XI.1 Add insertions to the vertical list . . . . .	135
XI.2 Penalties . . . . .	136
XI.3 Printing leftover notes . . . . .	137
<b>XII Critical footnotes</b>	<b>138</b>
XII.1 Fonts . . . . .	138
XII.2 Individual note options . . . . .	138
XII.3 Notes language . . . . .	139
XII.4 General survey of the way we manage notes . . . . .	140
XII.5 General setup . . . . .	140
XII.6 Footnotes arrangement . . . . .	141
XII.6.1 User level macro . . . . .	141
XII.6.2 Normal footnote . . . . .	142
XII.6.3 Paragraphed footnotes . . . . .	146
XII.6.4 Columnar footnotes . . . . .	153
XII.7 Critical notes presentation . . . . .	158
XII.7.1 Font tools . . . . .	158
XII.7.2 Pstart number in footnote . . . . .	159
XII.7.3 Line number printing . . . . .	159



<b>XIII Familiar footnotes</b>	<b>168</b>
XIII.1 Adjacent footnotes . . . . .	168
XIII.2 Regular footnotes for numbered texts . . . . .	169
XIII.3 Footnote formats . . . . .	170
XIII.4 Footnote arrangement . . . . .	170
XIII.4.1 User level macro . . . . .	170
XIII.4.2 Normal footnotes . . . . .	171
XIII.4.3 Two columns footnotes . . . . .	175
XIII.4.4 Three columns footnotes . . . . .	177
XIII.4.5 Paragraphed footnotes . . . . .	179
<b>XIV Code common to both critical and familiar footnote in normal arrangement</b>	<b>183</b>
<b>XV Footnotes' width for two columns</b>	<b>184</b>
<b>XVI Footnotes' order</b>	<b>185</b>
<b>XVII Footnotes' rule</b>	<b>185</b>
<b>XVIII Specific skip for first series of footnotes</b>	<b>186</b>
XVIII.0.1 Overview . . . . .	186
XVIII.0.2 User level command . . . . .	187
XVIII.0.3 Internal commands . . . . .	187
<b>XIX Endnotes</b>	<b>188</b>
<b>XX Generate series of notes</b>	<b>196</b>
XX.1 Test if series is still existing . . . . .	196
XX.2 Init specific to <code>reledpar</code> . . . . .	196
XX.3 For critical footnotes . . . . .	196
XX.3.1 Options . . . . .	197
XX.3.2 Create inserts, needed to add notes in foot . . . . .	198
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc. . . . .	198
XX.3.4 Set standard display . . . . .	199
XX.4 For familiar footnotes . . . . .	199
XX.4.1 Options . . . . .	200
XX.4.2 Create tools for familiar footnotes ( <code>\footnotex</code> ) . . . . .	200
XX.5 The endnotes . . . . .	201
XX.5.1 The auxiliary file . . . . .	201
XX.5.2 The main macro . . . . .	201
XX.5.3 The options . . . . .	202
XX.6 Init standards series (A,B,C,D,E) . . . . .	203



<b>XXI Setting series display</b>	<b>203</b>
XXI.1 Change series order . . . . .	203
XXI.2 Test series order . . . . .	204
XXI.2.1 Get the first series . . . . .	204
XXI.3 Series setting . . . . .	204
XXI.3.1 General way of working . . . . .	204
XXI.3.2 Tools to set options . . . . .	205
XXI.3.3 Tools to generate options commands . . . . .	206
XXI.3.4 Options for critical notes . . . . .	208
XXI.3.5 Options for familiar notes . . . . .	209
XXI.3.6 Options for endnotes . . . . .	210
XXI.4 Hooks for a particular footnote . . . . .	211
XXI.5 Alias . . . . .	212
<b>XXII Output routine</b>	<b>212</b>
XXII.0.1 Page number management . . . . .	212
XXII.0.2 Extra footnotes output . . . . .	212
XXII.0.3 Standard output's commands patching . . . . .	215
<b>XXIII Cross referencing</b>	<b>217</b>
<b>XXIV Side notes</b>	<b>226</b>
<b>XXV Minipages and such</b>	<b>232</b>
<b>XXVI Indexing</b>	<b>237</b>
<b>XXVII Verse</b>	<b>244</b>
XXVII.1 Hanging symbol management . . . . .	244
XXVII.2 Using & character . . . . .	244
XXVII.3 Code category setting . . . . .	245
XXVII.4 Stanza count and indent . . . . .	245
XXVII.5 Numbering stanza . . . . .	246
XXVII.6 Stanza number in note . . . . .	247
XXVII.7 Main work . . . . .	248
XXVII.8 Restore catcode and penalties . . . . .	250
<b>XXVIII Arrays and tables</b>	<b>250</b>
XXVIII.1 Preamble: macro as environment . . . . .	250
XXVIII.2 Tabular environments . . . . .	253
XXVIII.2.1 Disabling and restoring commands . . . . .	254
XXVIII.2.2 Counters, boxes and lengths . . . . .	257
XXVIII.2.3 Tabular typesetting . . . . .	261
XXVIII.2.4 Environments . . . . .	272
<b>XXIX Quotation's commands</b>	<b>272</b>



<b>XXX Section's title commands</b>	<b>273</b>
XXX.1 Commands to disable some feature . . . . .	273
XXX.2 General overview . . . . .	274
XXX.3 \beforeeledchapter command . . . . .	274
XXX.4 Auxiliary commands . . . . .	275
XXX.5 Patching standard commands . . . . .	276
XXX.6 Main code of \eledxxx commands . . . . .	280
XXX.7 Macros written in the auxiliary file . . . . .	283
<b>XXXI Page breaking or no page breaking depending of specific lines</b>	<b>285</b>
<b>XXXII Long verse: prevents being separated by a page break</b>	<b>287</b>
<b>XXXIII Compatibility with eledmac</b>	<b>287</b>
<b>Appendix A Some things to do when changing version</b>	<b>290</b>
Appendix A.1 Migrating from edmac to ledmac . . . . .	290
Appendix A.2 Migration from ledmac to eledmac . . . . .	291
Appendix A.3 Migration to eledmac 1.5.1 . . . . .	292
Appendix A.4 Migration to eledmac 1.12.0 . . . . .	292
Appendix A.5 Migration to eledmac 17.1 . . . . .	293
Appendix A.6 Migration to eledmac 1.21.0 . . . . .	293
Appendix A.6.1 \Xledsetnormalparstuffand\ledsetnormalparstuffX	293
Appendix A.6.2 Endnotes . . . . .	293
Appendix A.7 Migration to eledmac 1.22.0 . . . . .	293
Appendix A.8 Migration to eledmac 1.23.0 . . . . .	293
Appendix A.9 Migration from eledmac to reledmac . . . . .	294
Appendix A.9.1 Risk of 'no room for a new' . . . . .	294
Appendix A.9.2 Multiple indices with memoir . . . . .	294
Appendix A.9.3 Deprecated commands and options . . . . .	294
Appendix A.9.4 \renewcommandreplaced by command . . . . .	295
Appendix A.9.5 Commands the names of which have been changed . . .	295
Appendix A.9.6 Endnotes . . . . .	297
Appendix A.9.7 Z Series . . . . .	297
Appendix A.9.8 Internal commands . . . . .	297
Appendix A.10 Migration to reledmac 2.1.0 . . . . .	297
Appendix A.11 Migration to reledmac 2.1.3 . . . . .	297
Appendix A.12 Migration to reledmac 2.3.0 . . . . .	297
Appendix A.13 Migration to reledmac 2.4.0 . . . . .	298
<b>References</b>	<b>299</b>
<b>Index</b>	<b>299</b>
<b>Change History</b>	<b>341</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 description by John and Dominik of this version of `edmac` was published as 'An overview of `edmac`: a PLAIN  $\TeX$  format for critical editions', *TUGboat* 11 (1990), pp. 623–643.

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

As of 1994, we were pleased to be able to say that `edmac` was being used for the real-life book production of several interesting editions, such as the Latin texts of Euclid's *Elements*,<sup>2</sup> an edition of the letters of Nicolaus Copernicus,<sup>3</sup> Simon Bredon's *Arithmetica*,<sup>4</sup> a Latin translation by Plato of Tivoli of an Arabic astrolabe text,<sup>5</sup> a Latin translation of part II of the Arabic *Algebra* by Abū Kāmil Shujā' b. Aslam,<sup>6</sup> the Latin *Rithmarchia* of Werinher von Tegernsee,<sup>7</sup> a middle-Dutch romance epic on the Crusades,<sup>8</sup> a seventeenth-century Hungarian politico-philosophical tract,<sup>9</sup> an anonymous Latin compilation from Hun-

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

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

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

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

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

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

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

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



gary 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.

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. 291.

### 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. 294.

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

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

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



### 1.3 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  $\LaTeX$  itself. Although your textual apparatus and line numbers will be printed on the first run, it takes two more passes through  $\LaTeX$  to be sure that everything is correctly placed. If you make any subsequent changes altering the number of lines or notes, the input file may similarly require three passes to get everything to the right place. `reledmac` will tell you that you need to make more runs when it detects changes, but it does not expend the labor to check this thoroughly. If you have problems with a line or two misnumbered at the top of a page, try running  $\LaTeX$  once or twice more.

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-compat** help to migrate from `eledmac` to `reledmac` (see Appendix A.9.5 p. 295).

**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 15 p. 54).

**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 (12.3 p. 49).

**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 (14.2 p. 53).

**noend** disables tools for endnotes (`\Aendnote`, `\Bendnote` etc.). If you do not need endnotes, this option lets **eledmac** 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. 43). 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. 40) or `reledpar` (see 17.7 p. 58).

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

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

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

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

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

```
\setcounter{numberpstart}{value}
```

---

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



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 Lua $\TeX$  or Xe $\TeX$ , you must switch text direction *before* the `\pstart` command.

#### 4.2.7 Memory limits

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

`\pausenumbering`

`\resumenumbering`

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

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

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

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

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

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

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

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



### 4.3 Lineation commands

#### 4.3.1 Disabling lineation

`\numberlinefalse` 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 line numbers. For example:

`\linenumberlist`

`\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`  
`\endsub` 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.

#### 4.4.2 Locking lineation

`\startlock`  
`\endlock` 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. 40.

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



### 4.4.3 Setting and changing line number

`\setline`  
`\advanceline` 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`  
`\sublinenumberstyle` Line numbers are normally printed as arabic numbers. You can use `\linenumberstyle{<style>}` 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:

<pre>I am happy : I saw my friend \edtext{Smith}{ \Afootnote{Jones C, D.}} on Tuesday.</pre>	<pre>1 I am happy : I saw my friend Smith on 2 Tuesday.</pre>
--	---

---

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:



<pre> I am happy : \edtext{I saw my friend \edtext{Smith}{\Afootnote{Jones C, D.}} on Tuesday.}{ \Bfootnote{The date was July 16, 1954.} } </pre>	<pre> 1 I am happy : I saw my friend Smith on 2 Tuesday. </pre> <hr style="width: 100px; margin-left: 0;"/> <pre> 1 Smith ] Jones C, D. </pre> <hr style="width: 100px; margin-left: 0;"/> <pre> 1-2 I saw my friend Smith on Tuesday.] The date was July 16, 1954. </pre>
---	--

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

### 5.2.2 Footnotes

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

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

If you need more series of critical notes, please look at 5.5.1 p. 28.

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. 6.2.3 p. 31).
- `nonum` to disable line numbering for this note.
- `nosep` to disable the lemma separator for this note.
- `linerangesep=<c>` to change to `<c>` the separator between start line and end line for this particular note.

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

### 5.2.3 Endnotes

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

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

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

`\Aendnote[<option>]{<text>}`.  
`<option>` can contain a comma-separated list of values. Allowed values are:

- `fulllines` to disable `\Xendtwolines` and `\Xendmorethantwolines` features for this particular note (cf. 6.2.3 p. 31).



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

`\doendnotes`

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

`\doendnotesbysection`

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

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

Note that by default inside endnotes no separator is used between the lemma and the content. However you can use the `\Xendlemmaseparator` macro to define one (6.4.2 p. 34).

As endnotes may be printed at any point in the document they always start with the page number where they are called. The macro `\printnpnum{<num>}` is used to print these numbers. Its default definition is:

```
\newcommand*{\printnpnum}[1]{p.#1} }
```

#### 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. 44) 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.

#### 5.3.1 Basic use

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

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

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

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

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

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

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

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

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

- If you use `XYTeX`, add this line in your preamble: `\XeTeXinputnormalization 1`.

<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`.



- If you use Lua<sub>TeX</sub>, use the `uninormalize` package of Michal Hoftich<sup>15</sup> with the `buffer` option set to `true`.

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

### 5.3.3 Use with `\lemma` command

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

For example in the following example:

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

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

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

- In the content of `\lemma`, use `\sameword` with no optional argument.
- In the first argument of `\edtext`, use `\sameword` with the optional argument `[⟨X⟩]`. `⟨X⟩` is the depth of the `\edtext` where the `\lemma` is used. So if the `\lemma` is called in a `\edtext` inside another `\edtext`, `⟨X⟩` is equal to 2. If the `\lemma` is called in a `\edtext` “of first level”, `⟨X⟩` is equal to 1. If the lemma is called in both 1 and 2 `\edtext` depth, `⟨X⟩` is 1, 2. If that word is referenced in the lemma of every `\edtext` depth, `⟨X⟩` 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 `⟨X⟩` does not refer to the level where the `\sameword` occurs, but to the level of the `\lemma` that refers to that `\sameword`. For example:

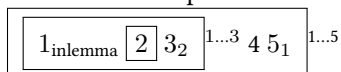
```
\edtext{some \edtext{\sameword[1]{word}}{\Afootnote{om. M}}
        and other \sameword{word}
        and again a \sameword{word}
        it is all}%
{\lemma{some \sameword{word} \ldots all}\Afootnote{critical note}}.%
```

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



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 `\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  $\TeX$  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 the place kept by the notes, including blank spaces.

When a length, noted *l*, is used, it can be stretchable: `a plus b minus c`. The final length *m* is calculated by  $\TeX$  to have:  $a - c \leq m \leq a + b$ . 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` By default, all footnotes are formatted as a series of separate paragraphs in one column.  
`\arrangementX` Three other formats are also available for notes.

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

The value of *a* 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 17.5 p. 57.
- `twocol` formats them as separate paragraphs, but in two columns;
- `threecol`, in three columns.
- `normal`, restore normal arrangement.

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

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



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!<sup>18</sup>

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  $\TeX$  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.

## 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`

`\Xendnumberonlyfirstintwolines`

`\Xendsymlinenum`

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

### 6.2.2 Separator for line range

`\Xlinerangeseparator`

`\Xendlinerangeseparator`

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

<sup>18</sup>See XII.6.3 p. 149 for further information.



### 6.2.3 Abbreviate line range

`\Xtwolines`  
`\Xmorethantwolines`

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)”.

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

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

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

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

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

It is possible to disable this again with `\Xtwolinesbutnotmore[⟨series⟩][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[⟨series⟩]`.

Use `\Xtwolinesonlyinsamepage[⟨series⟩][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. 22.

`\Xendtwolines`

For endnotes, use these macros: `\Xendtwolines`; `\Xendmorethantwolines`;

`\Xendtwolinesbutnotmore`;

`\Xendtwolinesonlyinsamepage` instead of `\Xtwolines`; `\Xmorethantwolines`;

`\Xtwolinesbutnotmore`; `\Xtwolinesonlyinsamepage`.

### 6.2.4 Disable line number

`\Xnonumber`  
`\Xendnonumber`

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

### 6.2.5 Printing pstart number

`\Xpstart`

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

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

### 6.2.6 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.

### 6.2.7 Space around number

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

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

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

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

### 6.2.8 Space around line symbol

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

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

`\XendbeforeSYMlinenum` `\XendafterSYMlinenum` are the equivalents of `\XbeforeSYMlinenum` and `\XafterSYMlinenum` for the endnotes.



### 6.2.9 Space in place of number

`\Xinplaceofnumber` If no number or symbolic line number is printed, you can add a space, with `\Xinplaceofnumber[\langle s \rangle]{\langle l \rangle}`. The default value is 1 em.

`\Xendingplaceofnumber` `\Xendingplaceofnumber[\langle s \rangle]{\langle l \rangle}` is the same, for critical endnotes.

### 6.2.10 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[\langle s \rangle]{\langle l \rangle}` 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[\langle s \rangle]{\langle l \rangle}` is the same as `\Xboxlinenum` but for the line number symbol.

`\Xendboxsymlinenum` `\Xendboxsymlinenum[\langle s \rangle]{\langle l \rangle}` 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[\langle s \rangle]{\langle text \rangle}` where `\langle text \rangle` 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[\langle s \rangle]{\langle l \rangle}` will box the start line number in a box of length `\langle l \rangle`. The content will be put at the right of the box.
- `\Xboxendlinenum[\langle s \rangle]{\langle l \rangle}` will box the dash plus the end line number or the range symbol in a box of length `\langle l \rangle`. 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[\langle s \rangle]{\langle l \rangle}`, `\Xendboxlinenumalign[\langle s \rangle]{\langle text \rangle}`, `\Xendboxstartlinenum[\langle s \rangle]{\langle l \rangle}`, `\Xendboxendlinenum[\langle s \rangle]{\langle l \rangle}` are the same as, respectively, `\Xboxlinenum` and `\Xboxendlinenum`, `\Xboxstartlinenum`, `\Xboxendlinenum` except in endnotes.

`\Xendboxlinenumalign`

`\Xendboxendlinenumalign`



### 6.3 Arbitrary code around line number

<code>\Xendbhooklinenumber</code>	<code>\Xendbhooklinenumber[⟨s⟩]{⟨code⟩}</code> is used to execute code before line number in endnotes. The code is executed before the <code>\Xendbeforelinenumber</code> space and before the <code>\Xendnotenumfont</code> font setting.
<code>\Xendahooklinenumber</code>	<code>\Xendahooklinenumber[⟨s⟩]{⟨code⟩}</code> is used to execute code after line number in endnotes. The code is executed after the <code>\Xendafternumber</code> space.
<code>\Xendbhookinplaceofnumber</code>	<code>\Xendbhookinplaceofnumber[⟨s⟩]{⟨code⟩}</code> is used to execute code before space or symbol which replace line number in endnotes. The code is executed before the <code>\Xendbeforelinenumber</code> space and before the <code>\Xendnotenumfont</code> font setting.
<code>\Xendahookinplaceofnumber</code>	<code>\Xendahookinplaceofnumber[⟨s⟩]{⟨code⟩}</code> is used to execute code after space or symbol which replace line number in endnotes. The code is executed after the <code>\Xendaftersynlinenum</code> space.

### 6.4 Separator between the lemma and the note

#### 6.4.1 For footnotes

<code>\Xlemmaseparator</code>	By default, in a footnote, the separator between the lemma and the note is a right bracket ( <code>\rbracket</code> ) <sup>19</sup> . You can use <code>\Xlemmaseparator[⟨s⟩]{⟨Xlemmaseparator⟩}</code> 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 <b>breakable</b> space between the separator and the following text.
<code>\Xbeforelemmaseparator</code>	Using <code>\Xbeforelemmaseparator[⟨s⟩]{⟨l⟩}</code> 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.
<code>\Xafterlemmaseparator</code>	Using <code>\Xafterlemmaseparator[⟨s⟩]{⟨l⟩}</code> 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.
<code>\Xnolemmaseparator</code>	You can suppress the lemma separator, using <code>\Xnolemmaseparator[⟨s⟩]</code> , which is simply a alias of <code>\Xlemmaseparator[⟨s⟩]{}</code> .
<code>\Xinplaceoflemmaseparator</code>	With <code>\Xinplaceoflemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add a space if no lemma separator is printed. The default value is 1 em.

#### 6.4.2 For endnotes

<code>\Xendlemmaseparator</code>	By default, there is no separator inside endnotes between the lemma and the content of the note. You can use <code>\Xendlemmaseparator[⟨s⟩]{⟨Xendlemmaseparator⟩}</code> to change this. The optional argument can be used to specify the series in which it is used. A common value of <code>⟨Xendlemmaseparator⟩</code> is <code>\rbracket</code> . Note that there is a non-breakable space between the lemma and the separator, but a <b>breakable</b> space between the separator and the following text.
<code>\Xendbeforelemmaseparator</code>	Using <code>\Xendbeforelemmaseparator[⟨s⟩]{⟨l⟩}</code> 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.
<code>\Xendafterlemmaseparator</code>	Using <code>\Xendafterlemmaseparator[⟨s⟩]{⟨l⟩}</code> you can add some space between

<sup>19</sup>For polyglossia, when the lemma is RTL, the bracket automatically switches to a left bracket.



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.

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

## 6.5 Font style

### 6.5.1 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`.

### 6.5.2 For the lemma

`\lemmadisablefontselection` 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.

`\lemmadisablefontselection` 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.

### 6.5.3 For all notes

`\Xnotefontsize` `\Xnotefontsize[⟨s⟩]{⟨command⟩}` is used to define the font size of critical footnotes of the series. The default value is `\footnotesize`. The `⟨command⟩` must not be a size in pt, but a standard  $\TeX$  size, like `\small`.

`\notefontsizeX` `\notefontsizeX[⟨s⟩]{⟨command⟩}` is used to define the font size of familiar footnotes of the series. The default value is `\footnotesize`. The `⟨command⟩` must not be a size in pt, but a standard  $\TeX$  size, like `\small`.

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

## 6.6 Indent of notes content

`\Xparindent` By default, `reledmac` does not add indentation before the paragraphs inside critical footnotes. Use `\Xparindent[⟨s⟩]` to enable indentation.

`\parindentX` By default, `reledmac` does not add indentation before the paragraphs inside familiar



footnotes. Use `\parindentX[⟨s⟩]` to enable indentation.

`\Xhangindent` For critical notes NOT paragraphed you can define an indent with `\Xhangindent[⟨s⟩]{⟨l⟩}`, 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.

`\hangindentX` For familiar notes NOT paragraphed you can define an indentation with `\hangindentX[⟨s⟩]{⟨l⟩}`, 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.

`\Xendhangindent` For critical endnotes NOT paragraphed you can define an indentation with `\Xendhangindent[⟨s⟩]{⟨l⟩}`, 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.

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

## 6.8 Options for footnotes in columns

### 6.8.1 Alignment

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

`<code>` must be one of the following command:

`\justifying` to have text justified, as usual with  $\text{\LaTeX}$ . 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`).



### 6.8.2 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> .

## 6.9 Options for paragraphed footnotes

### 6.9.1 Mark separation of notes

<code>\Xafternote</code>	You can add some 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.

### 6.9.2 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.

## 6.10 Options for block of notes

### 6.10.1 Text before notes

<code>\Xtxtbeforenotes</code>	You can add text before critical notes with <code>\Xtxtbeforenotes[⟨s⟩]{⟨text⟩}</code> .
-------------------------------	--

### 6.10.2 Spacing

<code>\Xbeforenotes</code>	You can change the vertical space before the rule of the critical notes with <code>\Xbeforenotes[⟨s⟩]{⟨l⟩}</code> . The default value is <code>1.2em plus .6em minus .6em</code> . <b>Be careful, the standard L<sup>A</sup>T<sub>E</sub>X footnote rule used by reledmac decreases by 3pt. This 3pt decrease is not changed by this command.</b>
<code>\beforenotesX</code>	You can change the vertical space printed before the rule of the familiar notes with <code>\beforenotesX[⟨s⟩]{⟨l⟩}</code> . The default value is <code>1.2em plus .6em minus .6em</code> .



**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{<l>}`. The default value is 0pt. You can disable this feature by setting the length to 0pt.

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

### 6.10.3 Rule

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

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

### 6.10.4 Maximum height

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

`\maxhnotesX` `\maxhnotesX[<s>]{<l>}` is the same as previous, but for familiar footnotes.  
Note that in many cases, you should call these commands, 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

```
\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 Footnotes and the `reledpar` columns

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

`noteswidthliketwocolumnsX`

- `\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.

## 6.12 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 an 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.

`\fullstop` A relatively trivial matter relates to punctuation. In your footnotes, there will be sometimes lines with sub-line numbers like this: 55.6. The en-dash and the full stop are taken from the same font as the numbers, and it all works nicely. But what if you wanted to use old-style numbers, like 12 and 34? These look nice in an edition, but when you use the fonts provided by `PLAIN TEX` they are taken from a math font which does not have full stop in the same places as a text font. If you (or your macros) just typed `${\oldstyle 55.6$}` you will get ‘55>6’. So we define `\fullstop`, which produces a full stop from the normal document font, whatever font you are using for the numbers. This macro is used in the macros which format the line numbers in the margins and footnotes, instead of explicit punctuation.

`\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.

`\setstanzaindent` In order to use the stanza macros, **one must set the indentation values**. First the value of `\stanzaindentbase` should be set, unless the default value 20pt is desired. Every stanza line indentation is a multiple of this.

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

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

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

If you want the hanging verse to be flush right, you can use `\sethanginsymbol`: see p. 8.6 p. 42.

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:



```
\setstanzaindents{5,1,0}
\setcounter{stanzaindentsrepetition}{2}
```

is like

```
\setstanzaindents{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. 292.**

If you don't use the `stanzaindentsrepetition` 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{stanzaindentsrepetition}{0}
```

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

## 8.4 Manual stanza indent

`\stanzaindent`  
`\stanzaindent*`

You can set the indent of some specific verse by calling `\stanzaindent{⟨value⟩}` at the beginning of the verse, before any other character. In this case, the indent defined by `\setstanzaindents` 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 16.2 p. 55 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`  
`\numberstanzafalse`

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

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

You can use this feature in combination with `\Xstanza` (6.2.6 p. 32).

`thestanza`

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



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

`\stanzanumwrapper`

You can change the value of the stanza counter with the usual commands of  $\TeX$ . 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. 42).

## 9 Grouping

In a `minipage` environment  $\TeX$  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}[⟨pos⟩]{⟨width⟩}`.



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{<lab>}`.  $\langle lab \rangle$  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>20</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` 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 Refer to a critical notes

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

The `\edtext{creature\edlabel{elephant}}` was quite

<sup>20</sup>More precisely, you should stick to characters in the  $\TeX$  categories of “letter” and “other”.



```
unafraid}{\Afootnote{Of the mouse, that is.}}
```

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

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

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

```
\xpageref
\xlineref
\xsublineref
\xpstartref
```

Where #1 stands for the reference.

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.
- When `hyperref` is loaded, the `hyperref` link will not be added. (Indeed, it is not a limitation, but a feature.)

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

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

## 10.6 References to lines commented 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  
`\apprefwithpage` lines. The first one will print the lines as they are printed in the critical footnotes, while the second will print the lines as they are printed in endnotes.

`\setapprefprefixsingle`      If you use `\setapprefprefixsingle{<prefix>}`, `<prefix>` will be printed before the  
`\setapprefprefixmore` line numbers of a `\appref`-reference. If you use `\setapprefprefixmore{<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.

`\Xtwolinesappref`      If you use `\Xtwolines`, `\Xmorethantwolines`, `\Xtwolinesbutnotmore` and/or  
`\Xmorethantwolinesappref` `\Xtwolinesonlyinsamepage` (6.2.3 p. 31) *without the optional series argument*, the setting will also be available for `\appref`.  
`\Xtwolinesbutnotmoreappref`  
`\Xtwolinesonlyinsamepage`

The commands `\Xtwolinesappref{<text>}`, `\Xmorethantwolinesappref{<text>}`, `\Xtwolinesbutnotmoreappref` `\Xtwolinesonlyinsamepageappref` can also be used, if you only want to change the reference style of `\appref`.

It is possible to disable this setting for a specific `\appref` command by using `\appref[fulllines]{<label>}`.

`\Xendtwolinesapprefwithpage`      If you use one of `\Xendtwolines`, `\Xendmorethantwolines`, `\Xendtwolinesbutnotmore`,  
`\Xendmorethantwolinesapprefwithpage` `\Xendtwolinesonlyinsamepage` (6.2.3 p. 31) *without the optional series argument*, the  
`\Xendtwolinesbutnotmoreapprefwithpage` setting will also be available for `\apprefwithpage`.  
`\Xendtwolinesonlyinsamepageapprefwithpage`



The commands `\Xendtwolinesappref{<text>}`, `\Xendmoreethantwolinesappref{<text>}`, `\Xendtwolinesbutnotmoreappref`, `\Xendtwolinesonlyinsamepageappref` can also be used, if you only want to change the reference style of `\apprefwithpage`.

It is possible to disable this setting for a specific `\apprefwithpage` command by using `\apprefwithpage[fulllines]{<label>}`.

## 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`  
`\ledsidenote`  
`\sidenotemargin`      The package's default setting is `\sidenotemargin{right}`

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

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

### 11.2 Setting

#### 11.2.1 Width

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

#### 11.2.2 Vertical position

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

#### 11.2.3 Distance to the main text

`\ledlsnotesep`      The texts are put a distance `\ledlsnotesep` (or `\ledrsnotesep`) into the left (or right)  
`\ledrsnotesep`      margin. These lengths are initially set to the value of `\linenumsep`.

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



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

These can of course be changed to suit.

#### 11.2.4 Separator between notes

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

## 12 Indexing

### 12.1 Basics

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

Note that the file `.idx` will contain the right reference only after the third run, because of the internal indexing mechanism of `eledmac`. 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 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>21</sup>.

<sup>21</sup>For further detail, you can read <http://tex.stackexchange.com/a/32783/7712>.



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

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

## 12.3 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>22</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>23</sup>

## 12.4 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.

<sup>22</sup>Or, for people who read French, read <http://geekographie.maieul.net/174>.

<sup>23</sup>These are the recommended lines to provide the best possible compatibility between hyperref and xindy, even without using reledmac.



## 13 Tabular material

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

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.

```

\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{edtabularr}
\pend
\endnumbering

```

produces the following parallel pair of verses.

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

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







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

`\edatleft`      `\edatleft[ $\langle math \rangle$ ]{ $\langle symbol \rangle$ }{ $\langle halfheight \rangle$ }` typesets the math  $\langle symbol \rangle$  as  $\left\{ \langle symbol \rangle \right\}$  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  $\right\{ \langle symbol \rangle \}$  with  $\langle math \rangle$  centered after it.

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

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

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

For example:

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

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

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

`\edvertdots`

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



```

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

```

$$\begin{array}{cccc}
 a & b & C & d \\
 v & w & x & y \\
 m & n & o & p \\
 k & & L & cvb
 \end{array}
 \left|
 \right.$$

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

## 14 Sectioning commands

### 14.1 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. 16):

```

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

```

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

### 14.2 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.

### 14.3 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.

## 15 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.

## 16 Page breaks

### 16.1 Control page breaking

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

```
\ledpb
\lednopb
```



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

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

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

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

## 16.2 Prevent page break in a long verses

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

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

## 17 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
```

## 17.1 Known and suspected limitations

### 17.2 ‘No room for a new’

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

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

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

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

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

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

### 17.3 Marginal notes

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

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

### 17.4 Paragraph shape

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

`\ballast`  $\LaTeX$  is a three-pass system, but even after a document has been processed three times, there are some tricky situations in which the page breaks decided by  $\TeX$  never settle down. At each successive run, `reledmac` may oscillate between two different sets of page decisions. To stop this happening, should it arise, Wayne Sullivan suggested the inclusion of the quantity `\ballast`. The amount of `\ballast` will be subtracted from the penalties which apply to the page breaks calculated on the *previous* run through  $\TeX$ , thus reinforcing these breaks. So if you find your page breaks oscillating, insert `\setcounter{ballast}{100}` or some such figure, and with any luck the page breaks will settle down. Luckily, this problem does not crop up at all often.



## 17.5 Paragraphed footnotes

The restriction on explicit line-breaking in paragraphed footnotes, mentioned in a footnote ?? p. ??, and described in more detail on XII.6.3 p. 149, 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` (6.10.4 p. 38), 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.

## 17.6 Use with other packages

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

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

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

If you actually try this<sup>24</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

<sup>24</sup>Reported by Dirk-Jan Dekker in the CTT thread ‘Incompatibility of “color” package’ on 2003/08/28.



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.

## 17.7 Parallel typesetting

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

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



## I Implementation overview

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

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

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

Next are commands for marking sections of the text for footnotes (Section VI), followed by the macros that take each paragraph apart, attach the line numbers and insertions, and send the result to the vertical list (Section VII). The footnote commands (Section XII) and output routine (Section ??) 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}[2015/10/19 v2.4.0 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","pardir","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 %

```

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

```

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

```

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

```

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

```

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.

```
103 \ifdef{\if@RTL}{-}{\newif\if@RTL}
104 %
```

## 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.

```
105 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
106 %
```

`\reledmac@error` Write an error message.

```
107 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
108 %
```

```
\led@err@NumberingStarted 109 \newcommand*\led@err@NumberingStarted{%
d@err@NumberingNotStarted 110 \reledmac@error{Numbering has already been started}{\@ehc}}
NumberingShouldHaveStarted 111 \newcommand*\led@err@NumberingNotStarted{%
112 \reledmac@error{Numbering was not started}{\@ehc}}
113 \newcommand*\led@err@NumberingShouldHaveStarted{%
114 \reledmac@error{Numbering should already have been started}{\@ehc}}
115 %
```

```
d@err@edtextoutsidestart 116 \newcommand*\led@err@edtextoutsidestart{%
117 \reledmac@error{\string\edtext\space outside numbered paragraph (\pstart\
ldots\pend)}{\@ehc}}%
118 %
```

```
\led@mess@NotesChanged 119 \newcommand*\led@mess@NotesChanged{%
120 \typeout{reledmac reminder: }%
121 \typeout{ The number of the footnotes in this section
122 has changed since the last run.}%
123 \typeout{ You will need to run LaTeX two more times
124 before the footnote placement}%
125 \typeout{ and line numbering in this section are
126 correct.}}
127 %
```



```

\led@mess@SectionContinued28 \newcommand*{\led@mess@SectionContinued}[1]{%
129 \message{Section #1 (continuing the previous section)}}
130 %

\led@err@LineationInNumbered31 \newcommand*{\led@err@LineationInNumbered}{%
132 \reledmac@error{You can't use \string\lineation\space within
133 a numbered section}{\@ehc}}
134 %

\led@warn@BadLineation35 \newcommand*{\led@warn@BadLineation}{%
\led@warn@BadLinenummargin36 \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp37 \newcommand*{\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp38 \reledmac@warning{Bad \string\linenummargin\space argument}}
139 \newcommand*{\led@warn@BadLockdisp}{%
140 \reledmac@warning{Bad \string\lockdisp\space argument}}
141 \newcommand*{\led@warn@BadSublockdisp}{%
142 \reledmac@warning{Bad \string\sublockdisp\space argument}}
143 %

\led@warn@NoLineFile44 \newcommand*{\led@warn@NoLineFile}[1]{%
145 \reledmac@warning{Can't find line-list file #1}}
146 %

\led@warn@LineFileObsolete47 \newcommand*{\led@warn@Obsolete}[1]{%
148 \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
149 %

\led@warn@BadAdvancelineSubline50 \newcommand*{\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine51 \reledmac@warning{\string\advanceline\space produced a sub-line
152 number less than zero.}}
153 \newcommand*{\led@warn@BadAdvancelineLine}{%
154 \reledmac@warning{\string\advanceline\space produced a line
155 number less than zero.}}
156 %

\led@warn@BadSetline57 \newcommand*{\led@warn@BadSetline}{%
\led@warn@BadSetlinenum58 \reledmac@warning{Bad \string\setline\space argument}}
159 \newcommand*{\led@warn@BadSetlinenum}{%
160 \reledmac@warning{Bad \string\setlinenum\space argument}}
161 %

```



```

\led@err@PstartNotNumbered62 \newcommand*{\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart63 \reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumbered64 numbered section}{\@ehc}}
\led@err@PendNoPstart65 \newcommand*{\led@err@PstartInPstart}{%
\led@err@AutoparNotNumbered66 \reledmac@error{\string\pstart\space encountered while another
\string\pstart\space was in effect}{\@ehc}}
\err@NumberingWithoutPstart67 \newcommand*{\led@err@PendNotNumbered}{%
168 \reledmac@error{\string\pend\space must be used within a
169 numbered section}{\@ehc}}
170 \newcommand*{\led@err@PendNoPstart}{%
171 \reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}
172 \newcommand*{\led@err@AutoparNotNumbered}{%
173 \reledmac@error{\string\autopar\space must be used within a
174 numbered section}{\@ehc}}
175 \newcommand*{\led@err@NumberingWithoutPstart}{%
176 \reledmac@error{\string\beginnumbering...\string\endnumbering\space
177 without \string\pstart}{\@ehc}}%
178 %

\led@warn@BadAction79 \newcommand*{\led@warn@BadAction}{%
180 \reledmac@warning{Bad action code, value \next@action.}}
181 %

\led@warn@DuplicateLabel82 \newcommand*{\led@warn@DuplicateLabel}[1]{%
\led@warn@AppLabelOutEdtext83 \reledmac@warning{Duplicate definition of label `#1' on page \the\pageno
\led@warn@RefUndefined .}}
184 \newcommand*{\led@warn@AppLabelOutEdtext}[1]{%
185 \reledmac@warning{\string\applabel\space outside of \string\edtext\space
`#1' on page \the\pageno.}}%
186 \newcommand*{\led@warn@RefUndefined}[1]{%
187 \reledmac@warning{Reference `#1' on page \the\pageno\space undefined.
188 Using `000'.}}
189 %

\led@warn@NoMarginpars90 \newcommand*{\led@warn@NoMarginpars}{%
191 \reledmac@warning{You can't use \string\marginpar\space in numbered text
}}
192 %

\led@warn@BadSidenotemargin93 \newcommand*{\led@warn@BadSidenotemargin}{%
194 \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
195 %

```



```

\led@warn@NoIndexFile96 \newcommand*{\led@warn@NoIndexFile}[1]{%
197 \reledmac@warning{Undefined index file #1}}
198 %

```

```

\led@warn@SeriesStillExist99 \newcommand{\led@warn@SeriesStillExist}[1]{%
200 \reledmac@warning{Series #1 is still existing !}%
201 }%
202 %

```

```

\led@err@ManySidenotes03 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes04 \ifledRcol{%
\led@err@ManyRightnotes05 \reledmac@warning{\itemcount@space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
206 \else%
207 \reledmac@warning{\itemcount@space sidenotes on line \the\line@num\
space p. \the\page@num}%
208 \fi%
209 }%
210 \newcommand{\led@err@ManyLeftnotes}{%
211 \ifledRcol{%
212 \reledmac@warning{\itemcount@space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
213 \else%
214 \reledmac@warning{\itemcount@space leftnotes on line \the\line@num\
space p. \the\page@num}%
215 \fi%
216 }%
217 \newcommand{\led@err@ManyRightnotes}{%
218 \ifledRcol{%
219 \reledmac@warning{\itemcount@space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
220 \else%
221 \reledmac@warning{\itemcount@space rightnotes on line \the\line@num\
space p. \the\page@num}%
222 \fi%
223 }%
224 %

```

```

\led@err@TooManyColumns25 \newcommand*{\led@err@TooManyColumns}{%
\led@err@UnequalColumns26 \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn27 \newcommand*{\led@err@UnequalColumns}{%
\led@err@HighEndColumn28 \reledmac@error{Number of columns is not equal to the number
\led@err@ReverseColumns29 in the previous row (or \protect\\space forgotten?)}{\
@ehc}}
230 \newcommand*{\led@err@LowStartColumn}{%
231 \reledmac@error{Start column is too low}{\@ehc}}

```



```

232 \newcommand*\led@err@HighEndColumn}{%
233   \reledmac@error{End column is too high}{\@ehc}}
234 \newcommand*\led@err@ReverseColumns}{%
235   \reledmac@error{Start column is greater than end column}{\@ehc}}
236 %

```

```

err@EdtextWithoutFootnote 37 \newcommand{\led@err@EdtextWithoutFootnote}{%
238   \reledmac@error{edtext without Xfootnote. Check syntaxis.}{\@ehc}%
239 }%
240 %

```

```

err@FootnoteWithoutEdtext 41 \newcommand{\led@err@FootnoteWithoutEdtext}{%
242   \reledmac@error{Xfootnote without edtext. Check syntax.}{\@ehc}%
243 }%
244 %

```

```

error@ImakeidxAfterEledmac 45 \newcommand{\led@error@ImakeidxAfterEledmac}{%
246   \reledmac@error{Imakeidx must be loaded before reledmac.}{\@ehc}%
247 }%
248 %

```

```

or@IndextoolsAfterEledmac 49 \newcommand{\led@error@IndextoolsAfterEledmac}{%
250   \reledmac@error{Indextools must be loaded before reledmac.}{\@ehc}%
251 }%
252 %

```

```

error@fail@patch@@makecol 53 \newcommand{\led@error@fail@patch@@makecol}{%
254   \reledmac@error{Fail to patch \string\@makecol\space command.}{\@ehc}%
255 }%
256 %

```

```

ror@fail@patch@@reinserts 57 \newcommand{\led@error@fail@patch@@reinserts}{%
258   \reledmac@error{Fail to patch \string\@reinserts\space command.}{\@ehc}%
259 }%
260 %

```

```

r@fail@patch@@doclearpage 61 \newcommand{\led@error@fail@patch@@doclearpage}{%
262   \reledmac@error{Fail to patch \string\@doclearpage\space command.}{\@ehc}
263   %
264 }%

```



```

\led@error@fail@patch@@iiiminipage 265 \newcommand{\led@error@fail@patch@@iiiminipage}{%
266 \reledmac@error{Fail to patch \string\@iiiminipage\space command.}\@ehc}
267 %
268 %

```

```

\led@error@fail@patch@endminipage 269 \newcommand{\led@error@fail@patch@endminipage}{%
270 \reledmac@error{Fail to patch \string\endminipage\space command.}\@ehc}%
271 }%
272 %

```

## II.8 Gobbling

Here, we define some commands which gobble their arguments.

```

\@gobblethree 273 \providecommand*\@gobblethree}[3]{}
\@gobblefour 274 \providecommand*\@gobblefour}[4]{}
\@gobblefive 275 \providecommand*\@gobblefive}[5]{}
276 %

```

## II.9 Miscellaneous commands

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

```

277 \ifledfinal
278 \newcommand*\showlemma[1]{#1}
279 \else
280 \newcommand*\showlemma[1]{\underline{#1}}
281 \fi
282
283 %

```

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

Initialize it as `\empty`.

```

284 \let\linenumberlist=\empty
285
286 %

```

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

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

```

287 \newcount\@l@tempcnta \newcount\@l@tempcntb
288 %

```



## II.10 Prepare reledpar

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.

<code>\ifnumberingR</code>	In preparation for the reledpar package, these are related to the ‘right’ text of parallel texts (when <code>\ifl@dpairing</code> is TRUE). They are explained in the <code>eledpar</code> manual.
<code>\ifl@dpairing</code>	
<code>\ifl@dpaging</code>	
<code>\l@dpagingtrue</code>	<code>\newif\ifl@dpairing</code>
<code>\l@dpagingfalse</code>	<code>\newif\ifl@dpaging%</code>
<code>\ifl@dprintingpages</code>	<code>\newif\ifl@dprintingpages%</code>
<code>\l@dprintingpagestrue</code>	<code>\newif\ifl@dprintingcolumns%</code>
<code>\l@dprintingpagesfalse</code>	<code>\newif\ifpst@rtedL</code>
<code>\ifl@dprintingcolumns</code>	<code>\newcount\l@dnumpstartsL</code>
<code>\l@dprintingcolumnstrue</code>	<code>%</code>
<code>\l@dprintingcolumnsfalse</code>	<code>\ifledRcol</code> is set to true in the Rightside environnement. It must be not confused with <code>\ifledRcol@</code> which is set to true when a right line is processed, in <code>\Pages</code> or <code>\Columns</code> .
<code>\l@dpairingtrue</code>	
<code>\l@dpairingfalse</code>	
<code>\ifpst@rtedL</code>	
<code>\pst@rtedLtrue</code>	<code>\newif\ifledRcol</code>
<code>\pst@rtedLfalse</code>	<code>\newif\ifledRcol@</code>
<code>\l@dnumpstartsL</code>	<code>%</code>
<code>\ifledRcol</code>	The <code>\ifnumberingR</code> flag is set to true if we’re within a right text numbered section.
<code>\ifledRcol@</code>	<code>\newif\ifnumberingR</code>
	<code>%</code>

### 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 `\resumenumbering` 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.

```
301 \newcount\section@num
302 \section@num=0
303 \let\extensionchars=\empty
304 %
```



`\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.

```
305 \newif\ifnumbering
306 %
```

`\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.

```
307 \newcommand*{\beginnumbering}{%
308   \ifnumbering
309     \led@err@NumberingStarted
310   \endnumbering
311 \fi
312 \global\numberingtrue
313 \global\advance\section@num \@ne
314 \initnumbering@reg
315 \message{Section \the\section@num }%
316 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
317 \l@dend@stuff
318 \setcounter{pstart}{1}
319 \ifl@dpairing
320   \global\l@dnumpstartsL \z@
321   \global\pst@rtedLfalse
322 %
```

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.



```

323 \else
324 \begingroup
325 \global\@afterindenttrue%In order to reestablish normal feature if the \
beginning was not here
326 \initnumbering@quote
327 \ifwidthliketwocolumns%
328 \csuse{setwidthliketwocolumns@\columns@position}%
329 \csuse{setpositionliketwocolumns@\columns@position}%
330 \fi%
331 \fi
332 \gdef\eled@sections@{}%
333 \if@noeled@sec\else%
334 \makeatletter\input{IfFileExists{\jobname.eledsec\the\section@num}{-}{-}\
makeatother}%
335 \immediate\openout\eled@sectioning@out=\jobname.eledsec\the\section@num
\relax%
336 \fi%
337 }
338 \newcommand*{\initnumbering@reg}{%
339 \global\pst@rtedLfalse
340 \global\l@dnumstartsL \z@
341 \global\absline@num \z@
342 \gdef\normal@page@break{}
343 \gdef\l@prev@pb{}
344 \gdef\l@prev@nopb{}
345 \global\line@num \z@
346 \global\subline@num \z@
347 \global\@lock \z@
348 \global\sub@lock \z@
349 \global\sublines@false
350 \global\let\next@page@num=\relax
351 \global\let\sub@change=\relax
352 \resetprevline@
353 \resetprevpage@num
354 }
355 %
356 %

```

**\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.

```

357 \def\endnumbering{%
358 \ifnumbering
359 \global\numberingfalse
360 \normal@pars
361 \ifnum\l@dnumstartsL=0%
362 \led@err@NumberingWithoutPstart%
363 \fi%
364 \ifl@dpairing

```



```

365 \global\pst@rtedLfalse
366 \else
367 \ifx\insertlines@list\empty\else
368 \global\noteschanged@true
369 \fi
370 \ifx\line@list\empty\else
371 \global\noteschanged@true
372 \fi
373 \fi
374 \ifnoteschanged@
375 \led@mess@NotesChanged
376 \fi
377 \else
378 \led@err@NumberingNotStarted
379 \fi
380 \autoparfalse
381 \if@noeled@sec\else%
382 \immediate\closeout\eled@sectioning@out%
383 \fi%
384 \ifl@dpairing\else
385 \global\l@dnumpststartsL=\z@%
386 \endgroup
387 \fi
388 }
389 %

```

`\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>25</sup>

`\resumenumbering`

```

390 \newcommand{\pausenumbering}{%
391 \ifautopar\global\autopar@pausettrue\fi%
392 \endnumbering\global\numberingtrue}
393 %

```

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.

```

394 \newcommand*{\resumenumbering}{%
395 \ifnumbering
396 \ifautopar@pause\autopar\fi
397 \global\pst@rtedLtrue
398 \global\advance\section@num \@ne
399 \led@mess@SectionContinued{\the\section@num}%
400 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
401 \l@dend@stuff
402 \ifl@dpairing\else%
403 \begingroup%

```

<sup>25</sup>Peter Wilson's thanks to Wayne Sullivan, who suggested the idea behind these macros.



```

404     \initnumbering@quote%
405     \ifwidthliketwocolumns%
406         \csuse{setwidthliketwocolumns@{\columns@position}}%
407         \csuse{setpositionliketwocolumns@{\columns@position}}%
408     \fi%
409 \fi%
410 \else
411     \led@err@NumberingShouldHaveStarted
412     \endnumbering
413     \beginnumbering
414 \fi}
415
416
417 %

```

## IV List macros

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

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

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

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

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

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

```

418 \newcommand*{\list@create}[1]{%
419     \global\let#1=\empty%
420 }%
421 %

```

`\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 .

```

422 \newcommand*{\list@clear}[1]{%
423     \global\let#1=\empty%
424 }
425 %

```



`\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`.

```

426 \newtoks\led@toksa \newtoks\led@toksb
427 \global\led@toksa={\}
428 \long\def\xright@appenditem#1\to#2{%
429   \global\led@toksb=\expandafter{#2}%
430   \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
431   \global\led@toksb={}}
432 %

```

`\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`.

```

433 \long\def\xleft@appenditem#1\to#2{%
434   \global\led@toksb=\expandafter{#2}%
435   \xdef#2{\the\led@toksa\expandafter{#1}\the\led@toksb}%
436   \global\led@toksb={}}
437 %

```

`\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.

```

438 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
439 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
440
441 %

```

## 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:  
`\bypstart@true`      • line-of-page: `bypstart@ = false` and `bypage@ = true`.  
`\bypstart@false`    • line-of-pstart: `bypstart@ = true` and `bypage@ = false`.  
`\ifbypage@`  
`\bypage@true`      reledmac will use the line-of-section system unless instructed otherwise.  
`\bypage@false`

```

442 \newif\ifbypage@
443 \newif\ifbypstart@
444 %

```



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@R45 \newif\ifbypage@R
\ifbypstart@R46 \newif\ifbypstart@R
447 %
```

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

```
448 \newcommand*{\lineation}[1]{%
449 %
```

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

```
450 \ifnumbering
451 \led@err@LineationInNumbered
452 \else
453 %
```

If the argument is `page`.

```
454 \def\@tempa{#1}\def\@tempb{page}%
455 \ifx\@tempa\@tempb
456 \global\bypage@true
457 \global\bypstart@false
458 \unless\ifnocritical@%
459 \Xpstart[] [false]%
460 \fi%
461 %
```

If the argument is `pstart`.

```
462 \else
463 \def\@tempb{pstart}%
464 \ifx\@tempa\@tempb
465 \global\bypage@false
466 \global\bypstart@true
467 \unless\ifnocritical@%
468 \Xpstart%
469 \fi%
470 %
```

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

```
471 \else
472 \def\@tempb{section}
473 \ifx\@tempa\@tempb
474 \global\bypage@false
475 \global\bypstart@false
476 \unless\ifnocritical@%
477 \Xpstart[] [false]%
```



```

478 \fi%
479 %

```

In other case, it is an error.

```

480 \else
481 \led@warn@BadLineation
482 \fi
483 \fi
484 \fi
485 \fi}}
486 %

```

## 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`  
`\l@getline@margin` The selection is recorded in the count `\line@margin`: 0 for left, 1 for right, 2 for outer, and 3 for inner.

```

487 \newcount\line@margin
488
489 \newcommand*{\linenummargin}[1]{%
490 \l@getline@margin{#1}%
491 \ifnum\@l@tempcntb>\m@ne
492 \ifledRcol
493 \global\line@marginR=\@l@tempcntb
494 \led@warn@setting@in@rightside{\linenummargin}%
495 \else
496 \global\line@margin=\@l@tempcntb
497 \fi
498 \fi}}
499
500 \newcommand*{\l@getline@margin}[1]{%
501 \def\@tempa{#1}\def\@tempb{left}%
502 \ifx\@tempa\@tempb
503 \@l@tempcntb \z@
504 \else
505 \def\@tempb{right}%
506 \ifx\@tempa\@tempb
507 \@l@tempcntb \@ne
508 \else
509 \def\@tempb{outer}%
510 \ifx\@tempa\@tempb
511 \@l@tempcntb \tw@
512 \else
513 \def\@tempb{inner}%
514 \ifx\@tempa\@tempb
515 \@l@tempcntb \thr@@
516 \else

```



```

517         \led@warn@BadLinenummargin
518         \@l@tempcntb \m@ne
519         \fi
520     \fi
521 \fi
522 \fi}
523
524 %

```

### 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.

```

525 \newcounter{firstlinenum}
526 \setcounter{firstlinenum}{5}
527 \newcounter{linenumincrement}
528 \setcounter{linenumincrement}{5}
529 %

```

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

```

530 \newcounter{firstsublinenum}
531 \setcounter{firstsublinenum}{5}
532 \newcounter{sublinenumincrement}
533 \setcounter{sublinenumincrement}{5}
534
535 %

```

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

```

536 \newcommand*{\firstlinenum}[1]{%
537     \ifledRcol%
538         \setcounter{firstlinenumR}{#1}%
539         \led@warn@setting@in@rightside{\firstlinenum}
540     \else%
541         \setcounter{firstlinenum}{#1}%
542     \fi%
543 }
544 \newcommand*{\linenumincrement}[1]{%
545     \ifledRcol%
546         \setcounter{linenumincrementR}{#1}%
547         \led@warn@setting@in@rightside{\linenumincrement}
548     \else%
549

```



```

550 \setcounter{linenumincrement}{#1}%
551 \fi%
552 }
553 \newcommand*\firstsublinenum}[1]{%
554 \ifledRcol%
555 \setcounter{firstsublinenumR}{#1}%
556 \led@warn@setting@in@rightside{\firstsublinenum}
557 \else%
558 \setcounter{firstsublinenum}{#1}%
559 \fi%
560 }
561 \newcommand*\sublinenumincrement}[1]{%
562 \ifledRcol%
563 \setcounter{sublinenumincrementR}{#1}%
564 \led@warn@setting@in@rightside{\sublinenumincrement}
565 \else%
566 \setcounter{sublinenumincrement}{#1}%
567 \fi%
568 }
569 %
570 %

```

## 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.

```

571 \newcount\lock@disp
572 \newcommand{\lockdisp}[1]{%
573 \l@dgetlock@disp{#1}%
574 \ifnum\l@dtempcntb>\m@ne
575 \global\lock@disp=\l@dtempcntb
576 \else
577 \led@warn@BadLockdisp
578 \fi}}
579 \newcommand*\l@dgetlock@disp}[1]{
580 \def\@tempa{#1}\def\@tempb{first}%
581 \ifx\@tempa\@tempb
582 \l@dtempcntb \z@
583 \else
584 \def\@tempb{last}%
585 \ifx\@tempa\@tempb
586 \l@dtempcntb \@ne
587 \else
588 \def\@tempb{all}%
589 \ifx\@tempa\@tempb

```



```

590     \l@dttempcntb \tw@
591     \else
592     \l@dttempcntb \m@ne
593     \fi
594 \fi
595 \fi}
596
597 %

```

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

```

598 \newcount\sublock@disp
599 \newcommand{\sublockdisp}[1]{\{%
600   \l@dgetlock@disp{#1}%
601   \ifnum\l@dttempcntb>\m@ne
602     \global\sublock@disp=\l@dttempcntb
603   \else
604     \led@warn@BadSublockdisp
605   \fi}}
606
607 %

```

## V.5 Line number style

`\linenumberstyle` We provide a mechanism for using different representations of the line numbers, not  
`\linenumrep` just the normal arabic.  
`\linenumr@p` NOTE: In v0.7 `\linenumrep` and `\sublinenumrep` replaced the internal `\linenumr@p`  
`\sublinenumberstyle` and `\sublinenumr@p`.  
`\sublinenumrep` `\linenumberstyle` and `\sublinenumberstyle` are user level macros for setting  
`\sublinenumr@p` the number representation (`\linenumrep` and `\sublinenumrep`) for line and sub-line  
 numbers.

```

608 \newcommand*{\linenumberstyle}[1]{\%
609   \def\linenumrep##1{\@nameuse{@#1}{##1}}
610 \newcommand*{\sublinenumberstyle}[1]{\%
611   \def\sublinenumrep##1{\@nameuse{@#1}{##1}}
612 %

```

Initialise the number styles to arabic.

```

613 \linenumberstyle{arabic}
614 \let\linenumr@p\linenumrep
615 \sublinenumberstyle{arabic}
616 \let\sublinenumr@p\sublinenumrep
617
618 %

```



## V.6 Line number printing

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

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

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

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

```

619 \newlength{\linenumsep}
620 \setlength{\linenumsep}{1pc}
621 \newcommand*{\numlabfont}{\normalfont\scriptsize}
622 \newcommand*{\ledlinenum}{%
623   \bgroup%
624   \ifluatex%
625     \texdir TLT%
626   \fi%
627   \numlabfont\linenumrep{\line@num}%
628   \ifsublines@
629     \ifnum\subline@num>0\relax
630       \unskip\fullstop\sublinenumrep{\subline@num}%
631     \fi
632   \fi%
633   \egroup%
634 }%
635
636 \newcommand*{\leftlinenum}{%
637   \ledlinenum
638   \kern\linenumsep}
639 \newcommand*{\rightlinenum}{%
640   \kern\linenumsep
641   \ledlinenum}
642
643 %

```

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

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

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

```
644 \newcount\line@num
645 %
```

`\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.

```
646 \newcount\subline@num
647 %
```

`\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.

`\sublines@false`

```
648 \newif\ifsublines@
649 %
```

`\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.

```
650 \newcount\absline@num
651 %
```

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.

```
652 \newcount\@lock
653 \newcount\sub@lock
654 %
```

## V.9 Line number associated to lemma

`\line@list` Now we can define the list macros that will be created from the line-list file. We will maintain the following lists:

`\insertlines@list`  
`\actionlines@list`  
`\actions@list`

- `\line@list`: the page and line numbers for every lemma marked by `\edtext`. There are seven pieces of information, separated by vertical bars:
  1. the starting page,
  2. line, and
  3. sub-line numbers, followed by the
  4. ending page,
  5. line, and
  6. sub-line numbers, and then the
  7. font specifier for the lemma.

These line numbers are all visible numbers. The font specifier is a set of four codes for font encoding, family, series, and shape, separated by / characters. Thus a lemma that started on page 23, line 35 and went on until page 24, line 3 (with no sub-line numbering), and was typeset in a normal roman font would have a line list entry like this:

```
23|35|0|24|3|0|OT1/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:

```
655 \list@create{\line@list}
656 \list@create{\insertlines@list}
657 \list@create{\actionlines@list}
658 \list@create{\actions@list}
659
660 %
```

`\page@num` 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.

```
\endpage@num
\endline@num
\endsubline@num
661 \newcount\page@num
662 \newcount\endpage@num
663 \newcount\endline@num
664 \newcount\endsubline@num
665 %
```

`\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 `ℒTEX`, 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 `ℒTEX` 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.

```
666 \newif\ifnoteschanged@
667 %
```

`\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@ 668 \newcommand*{\resetprevline@}{%
669 \def\do##1{\global\csundef{prevline##1}}}%
670 \dolistloop{\@series}%
671 }
672 %
```



`\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@  \newcommand*{\resetprevpage@num}{%
673
674   \def\do##1{\ifcsdef{prevpage##1@num}{\global\csname prevpage##1@num\
endcsname=0}{}}%
675   \dolistloop{\@series}%
676 }
677 %

```

## 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.

```

678 \newread\@inputcheck
679 \newcommand*{\read@linelist}[1]{%
680   \ifledRcol%
681     \list@clearing@regR%
682   \else%
683     \list@clearing@reg%
684   \fi%
685 %

```

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).

```

686   \list@clear{\maxlinesinpar@list}
687 %

```

Now get the file and interpret it. When the file is there we start a new group and make some special definitions we will need to process it. It is a sequence of  $\TeX$  commands, but they require a few special settings. We make `[` and `]` become grouping characters: they are used that way in the line-list file, because we need to write them out one at a time rather than in balanced pairs, and it is easier to just use something other than real braces. `@` must become a letter, since this is run in the ordinary  $\LaTeX$  context. We ignore carriage returns, since if we are in horizontal mode they can get interpreted as spaces to be printed.

Our line, page, and line-locking counters were already zeroed by `\line@list@stuff` if this is being called from within `\beginnumbering`; sub-lineation will be turned off as well in that case. On the other hand, if this is being called from `\resumenumbers`, those things should still have the values they had when `\pausenumbering` was executed.

If the file is not there, we print an informative message.

Now, after these preliminaries, we start interpreting the file.



```

688 \get@linelistfile{#1}%
689 \endgroup
690 %

```

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.

```

691 \ifledRcol
692   \global\page@numR=\m@ne
693   \ifx\actionlines@listR\empty
694     \gdef\next@actionlineR{1000000}%
695   \else
696     \glp\actionlines@listR\to\next@actionlineR
697     \glp\actions@listR\to\next@actionR
698   \fi
699 \else
700   \global\page@num=\m@ne
701   \ifx\actionlines@list\empty
702     \gdef\next@actionline{1000000}%
703   \else
704     \glp\actionlines@list\to\next@actionline
705     \glp\actions@list\to\next@action
706   \fi
707 \fi
708 }
709 %

```

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

```

710 \newcommand*{\list@clearing@reg}{%
711   \list@clear{\line@list}%
712   \list@clear{\insertlines@list}%
713   \list@clear{\actionlines@list}%
714   \list@clear{\actions@list}%
715   \list@clear{\linesinpar@listL}%
716   \list@clear{\linesonpage@listL}%
717 }%
718 %

```

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

```

719 \newcommand*{\get@linelistfile}[1]{%
720   \InputIfFileExists{#1}{%
721     \global\noteschanged@false
722     \begingroup
723     \catcode`\[=1 \catcode`\]=2
724     \makeatletter \catcode`\^M=9}{%

```



```

725 \led@warn@NoLineFile{#1}%
726 \global\noteschanged@true
727 \begingroup}%
728 }
729
730 %

```

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.

```

731 \newcommand{\line@list@version}[1]{%
732   \IfStrEq{#1}{\this@line@list@version}%
733   {}%
734   {\ifledRcol%
735     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
736     \else%
737     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
738     \fi%
739   \endinput%
740   }%
741 }%
742 %

```



`\@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`.

```

743
744 \newcommand*{\@nl}[2]{%
745   \fix@page{#1}%
746   \ifledRcol%
747     \@nl@regR%
748   \else%
749     \@nl@reg%
750   \fi%
751 }
752 \newcommand*{\@nl@reg}{%
753   \ifx\l@dchset@num\relax \else
754     \advance\absline@num \@ne
755     \set@line@action
756     \let\l@dchset@num=\relax
757     \advance\absline@num \m@ne
758     \advance\line@num \m@ne
759   \fi
760 %

```

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

```

761   \advance\absline@num \@ne
762   \ifx\next@page@num\relax \else
763     \page@action
764     \let\next@page@num=\relax
765   \fi
766   \ifx\sub@change\relax \else
767     \ifnum\sub@change>\z@
768       \sublines@true
769     \else
770       \sublines@false
771     \fi
772   \sub@action
773   \let\sub@change=\relax

```



```

774 \fi
775 %

```

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

```

776 \ifcase\@lock
777 \or
778 \@lock \tw@
779 \or \or
780 \@lock \z@
781 \fi
782 \ifcase\sub@lock
783 \or
784 \sub@lock \tw@
785 \or \or
786 \sub@lock \z@
787 \fi
788 %

```

Now advance the visible line number, unless it has been locked.

```

789 \ifsublines@
790 \ifnum\sub@lock<\tw@
791 \advance\subline@num \@ne
792 \fi
793 \else
794 \ifnum\@lock<\tw@
795 \advance\line@num \@ne \subline@num \z@
796 \fi
797 \fi}
798 %
799 %

```

`\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`.

```

800 \newcount\last@page@num
801 \last@page@num=-10000
802
803 \newcommand*{\fix@page}[1]{%
804 \iflabeledRcol
805 \ifnum #1=\last@page@numR
806 \else
807 \ifbypage@R
808 \line@numR \z@ \subline@numR \z@
809 \fi
810 \page@numR=#1\relax
811 \last@page@numR=#1\relax
812 \def\next@page@numR{#1}%
813 \fi
814 \else

```



```

815 \ifnum #1=\last@page@num
816 \else
817 \ifbypage@
818 \line@num \z@ \subline@num \z@
819 \fi
820 \page@num=#1\relax
821 \last@page@num=#1\relax
822 \def\next@page@num{#1}%
823 \listxadd{normal@page@break}{\the\absline@num}
824 \fi
825 \fi}
826 %

```

**\@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.

```

827 \@lopR \newcommand*{\@pend}[1]{}
828 \newcommand*{\@pendR}[1]{}
829 \newcommand*{\@lopL}[1]{}
830 \newcommand*{\@lopR}[1]{}
831
832 %

```

**\sub@on** The `\sub@on` and `\sub@off` macros turn sub-lineation on and off: but not directly, since  
**\sub@off** 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.

```

833 \newcommand*{\sub@on}{\ifsublines@
834 \let\sub@change=\relax
835 \else
836 \def\sub@change{1}%
837 \fi}
838 \newcommand*{\sub@off}{\ifsublines@
839 \def\sub@change{-1}%
840 \else
841 \let\sub@change=\relax
842 \fi}
843
844 %

```

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

```

845
846 \newcommand*{\@adv}[1]{%
847 \ifsublines@
848 \ifledRcol
849 \advance\subline@numR by #1\relax
850 \ifnum\subline@numR<\z@

```



```

851         \led@warn@BadAdvancelineSubline
852         \subline@numR \z@
853     \fi
854 \else
855     \advance\subline@num by #1\relax
856     \ifnum\subline@num<\z@
857         \led@warn@BadAdvancelineSubline
858         \subline@num \z@
859     \fi
860 \fi
861 \else
862     \ifledRcol
863         \advance\line@numR by #1\relax
864         \ifnum\line@numR<\z@
865             \led@warn@BadAdvancelineLine
866             \line@numR \z@
867         \fi
868     \else
869         \advance\line@num by #1\relax
870         \ifnum\line@num<\z@
871             \led@warn@BadAdvancelineLine
872             \line@num \z@
873         \fi
874     \fi
875 \fi
876 \set@line@action}
877
878 %

```

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

```

879
880 \newcommand*{\@set}[1]{%
881     \ifledRcol
882         \ifsublines@
883             \subline@numR=#1\relax
884         \else
885             \line@numR=#1\relax
886         \fi
887         \set@line@action
888     \else
889         \ifsublines@
890             \subline@num=#1\relax
891         \else
892             \line@num=#1\relax
893         \fi
894         \set@line@action
895     \fi}
896

```



897 %

**\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.

```
898
899 \newcommand*{\l@d@set}[1]{%
900   \ifl@dchset@num
901     \line@numR=#1\relax
902     \advance\line@numR \@ne
903     \def\l@dchset@num{#1}
904   \else
905     \line@num=#1\relax
906     \advance\line@num \@ne
907     \def\l@dchset@num{#1}
908   \fi}
909 \let\l@dchset@num\relax
910
911 %
```

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

```
912
913 \newcommand*{\page@action}{%
914   \ifl@dchset@num
915     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
916     \xright@appenditem{\next@page@numR}\to\actions@listR
917   \else
918     \xright@appenditem{\the\absline@num}\to\actionlines@list
919     \xright@appenditem{\next@page@num}\to\actions@list
920   \fi}
921 %
```

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

```
922
923 \newcommand*{\set@line@action}{%
924   \ifl@dchset@num
925     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
926     \ifsublines@
927       \@l@tempcnta=-\subline@numR
928     \else
929       \@l@tempcnta=-\line@numR
930     \fi
931     \advance\@l@tempcnta by -5000\relax
932     \xright@appenditem{\the\@l@tempcnta}\to\actions@listR
933   \else
```



```

934 \xright@appenditem{\the\absline@num}\to\actionlines@list
935 \ifsublines@
936 \@l@dttempcnta=-\subline@num
937 \else
938 \@l@dttempcnta=-\line@num
939 \fi
940 \advance\@l@dttempcnta by -5000\relax
941 \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
942 \fi}
943 %

```

**\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.

```

944
945 \newcommand*{\sub@action}{%
946 \ifledRcol
947 \xright@appenditem{\the\absline@numR}\to\actionlines@listR
948 \ifsublines@
949 \xright@appenditem{-1001}\to\actions@listR
950 \else
951 \xright@appenditem{-1002}\to\actions@listR
952 \fi
953 \else
954 \xright@appenditem{\the\absline@num}\to\actionlines@list
955 \ifsublines@
956 \xright@appenditem{-1001}\to\actions@list
957 \else
958 \xright@appenditem{-1002}\to\actions@list
959 \fi
960 \fi}
961 %

```

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

Adding commands to the action list is slow, and it is very often the case that a lock-on command is immediately followed by a lock-off command in the line-list file, and therefore really does nothing. We use a look-ahead scheme here to detect such pairs, and add nothing to the line-list in those cases.

```

962 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
963
964 \newcommand*{\do@lockon}{%
965 \ifx\next\lock@off
966 \global\let\lock@off=\skip@lockoff
967 \else
968 \ifledRcol
969 \do@lockonR

```



```

970 \else
971 \do@lockonL
972 \fi
973 \fi}
974
975
976 \newcommand*{\do@lockonL}{%
977 \xright@appenditem{\the\absline@num}\to\actionlines@list
978 \ifsublines@
979 \xright@appenditem{-1005}\to\actions@list
980 \ifnum\sub@lock=\z@
981 \sub@lock \@ne
982 \else
983 \ifnum\sub@lock=\thr@@
984 \sub@lock \@ne
985 \fi
986 \fi
987 \else
988 \xright@appenditem{-1003}\to\actions@list
989 \ifnum\@lock=\z@
990 \@lock \@ne
991 \else
992 \ifnum\@lock=\thr@@
993 \@lock \@ne
994 \fi
995 \fi
996 \fi}
997
998 %

```

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

```

\do@lockoff
\do@lockoffL
\skip@lockoff
999 \newcommand*{\do@lockoffL}{%
1000 \xright@appenditem{\the\absline@num}\to\actionlines@list
1001 \ifsublines@
1002 \xright@appenditem{-1006}\to\actions@list
1003 \ifnum\sub@lock=\tw@
1004 \sub@lock \thr@@
1005 \else
1006 \sub@lock \z@
1007 \fi
1008 \else
1009 \xright@appenditem{-1004}\to\actions@list
1010 \ifnum\@lock=\tw@
1011 \@lock \thr@@
1012 \else
1013 \@lock \z@
1014 \fi
1015 \fi}
1016

```



```

1017 \newcommand*{\do@lockoff}{%
1018   \ifledRcol
1019     \do@lockoffR
1020   \else
1021     \do@lockoffL
1022   \fi}
1023 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}
1024 \global\let\lock@off=\do@lockoff
1025
1026 %

```

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

```

1027 \newcommand*{\n@num}{%
1028   \ifledRcol%
1029     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1030     \xright@appenditem{-1007}\to\actions@listR
1031   \else%
1032     \xright@appenditem{\the\absline@num}\to\actionlines@list%
1033     \xright@appenditem{-1007}\to\actions@list%
1034   \fi%
1035 }%
1036
1037 %

```

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

```

1038 \newcommand*{\n@num@stanza}{%
1039   \ifledRcol%
1040     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1041     \xright@appenditem{-1008}\to\actions@listR%
1042   \else%
1043     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1044     \xright@appenditem{-1008}\to\actions@list%
1045   \fi%
1046 }
1047 %

```

**\ifl@dhiddenumber** `\hidenumbering` hides number in margin. It uses action code 1009.

**\hidenumbering**

```

1048 \newif\ifl@dhiddenumber
1049 \newcommand*{\hidenumbering}{
1050   \ifledRcol%
1051     \write\linenum@outR{\string\hide@num}%
1052   \else%
1053     \write\linenum@out{\string\hide@num}%
1054   \fi%

```



```

1055 }%
1056 \newcommand*\hide@num{%
1057   \ifledRcol%
1058     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1059     \xright@appenditem{-1009}\to\actions@listR%
1060   \else%
1061     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1062     \xright@appenditem{-1009}\to\actions@list%
1063   \fi%
1064 }
1065 %

```

**\@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.

```

1066 \newcount\insert@count
1067 %

```

- #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.

```

1068 \newcommand*\dummy@ref}[2]{#2}
1069 %

```

**\@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.

```

1070 \newcommand*\@ref}[2]{%
1071   \ifledRcol%
1072     \@ref@regR{#1}{#2}%
1073   \else%
1074     \@ref@reg{#1}{#2}%
1075   \fi%
1076 }%
1077 \newcommand*\@ref@reg}[2]{%
1078   \global\insert@count=#1\relax
1079   \global\advance\@edtext@level by 1%
1080   \loop\ifnum\insert@count>\z@
1081     \xright@appenditem{\the\absline@num}\to\insertlines@list
1082     \global\advance\insert@count \m@ne
1083   \repeat
1084 %

```



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.

```

1085 \begingroup
1086   \let\@ref=\dummy@ref
1087   \let\@lopL\@gobble
1088   \let\page@action=\relax
1089   \let\sub@action=\relax
1090   \let\set@line@action=\relax
1091   \let\@lab=\relax
1092   \let\@lemma=\relax%
1093   \let\@sw\@gobblethree%
1094   #2
1095   \global\endpage@num=\page@num
1096   \global\endline@num=\line@num
1097   \global\endsubline@num=\subline@num
1098 \endgroup
1099 %

```

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

```

1100   \xright@appenditem%
1101     {\the\page@num|\the\line@num}%
1102     \ifsublines@ \the\subline@num \else 0\fi}%
1103     \the\endpage@num|\the\endline@num}%
1104     \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1105 %

```

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.

```

1106   \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
1107   @edtext@level\endcsname}%
1107   \providebool{lemmacommand@\the\@edtext@level}%
1108   \boolfalse{lemmacommand@\the\@edtext@level}%
1109 %

```

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

```

1110   #2%
1111 %

```

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.

```

1112   \ifnum\@edtext@level>0%
1113     \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
1114     csname sw@list@edtext@\the\@edtext@level\endcsname}}}%
1114   \ifcsundef{sw@list@edtext@\the\@edtext@level}{\create@this@edtext@level
    }{}%

```



```

1115 \letcs{\@tmp}{sw@list@edtext@the\@edtext@level}%
1116 \letcs{\@tmpp}{sw@list@edtext@tmp@the\@edtext@level}
1117 \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
1118 \global\cslet{sw@list@edtext@the\@edtext@level}{\@tmp}%
1119 \fi%
1120 %

Decrease edtext level counter.

1121 \global\advance\@edtext@level by -1%
1122 %

1123 }
1124
1125 %

```

## 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`.

```

1126 \newwrite\linenum@out
1127 %

```

`\iffirst@linenum@out@` Once any file is opened on this stream, we keep it open forever, or else switch to another file that we keep open. The reason is that we want the output routine to write the page number for every page to this file; otherwise we would have to write it at the start of every line. But it is not very easy for the output routine to tell whether an output stream is open or not. There is no way to test the status of a particular output stream directly, and the asynchronous nature of output routines makes the status hard to determine by other means.

We can manage pretty well by means of the `\iffirst@linenum@out@` flag; its inelegant name suggests the nature of the problem that made its creation necessary. It is set to be true before any `\linenum@out` file is opened. When such a file is opened for the first time, it is done using `\immediate`, so that it will at once be safe for the output routine to write to it; we then set this flag to false.

```

1128 \newif\iffirst@linenum@out@
1129 \iffirst@linenum@out@true
1130 %

```

`\this@line@list@version` The commands allowed in the line-list file and their arguments can change between two versions of `reledmac`. The `\this@line@list@version` command is upgraded when it happens. It is written in the file list. If we process a line-list file which used an older version, that means the commands used inside are deprecated, and we can't use them.



```

1131 \newcommand{\this@line@list@version}{3}%
1132 %

```

**\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.

```

1133 \newcommand*{\line@list@stuff}[1]{%
1134 %

```

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

```

1135 \read@linelist{#1}%
1136 %

```

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.

```

1137 \iffirst@linenum@out@
1138 \immediate\closeout\linenum@out%
1139 \global\first@linenum@out@false%
1140 \immediate\openout\linenum@out=#1\relax%
1141 \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}%
1142 \ifl@dpaging%
1143 \immediate\write\linenum@out{\string\@par@sync@option{\
@par@this@sync@option}}%
1144 \fi%
1145 \else
1146 %

```

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.

```

1147 \if@minipage%
1148 \leavevmode%
1149 \fi%
1150 \closeout\linenum@out%
1151 \openout\linenum@out=#1\relax%
1152 \fi}
1153
1154 %

```

**\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.

```

1155 \newcommand*{\new@line}{%
1156 \IfStrEq{\led@pb@setting}{after}%
1157 {\xifinlist{\the\absline@num}{\l@prev@nopb}%
1158 {\xifinlist{\the\absline@num}{\normal@page@break}%

```



```

1159     {\numgdef{\@next@page}{\c@page+1}%
1160     \write\linenum@out{\string\@nl[\@next@page][\@next@page]}}%
1161     }%
1162     {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1163     }%
1164     {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1165     {}%
1166     \IfStrEq{\led@pb@setting}{before}%
1167     {\numdef{\next@absline}{\the\absline@num+1}%
1168     \xifinlist{\next@absline}{\l@prev@nopb}%
1169     {\xifinlist{\the\absline@num}{\normal@page@break}%
1170     {\numgdef{\nc@page}{\c@page+1}%
1171     \write\linenum@out{\string\@nl[\nc@page][\nc@page]}}%
1172     }%
1173     {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1174     }%
1175     {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}}%
1176     }%
1177     {}%
1178     \IfStrEqCase{\led@pb@setting}{before}{\relax}{after}{\relax}}{\write\
linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1179 }
1180
1181 %

```

**\if@noneed@Footnote** \if@noneed@Footnote is a boolean to check if we have to print a error message when a \edtext is called without any critical notes.

**\flag@start** We enclose a lemma marked by \edtext in \flag@start and \flag@end: these send the \@ref command to the line-list file. \edtext is responsible for setting the value of \insert@count appropriately; it actually gets done by the various footnote macros.

**\flag@end**

```

1182 \newif\if@noneed@Footnote%
1183
1184 \newcommand*{\flag@start}{%
1185   \ifledRcol%
1186     \edef\next{\write\linenum@outR{%
1187       \string\@ref[\the\insert@countR] []}}%
1188     \next%
1189     \ifnum\insert@countR<1%
1190       \if@noneed@Footnote\else%
1191         \led@err@EdtextWithoutFootnote%
1192       \fi%
1193     \fi%
1194   \else%
1195     \edef\next{\write\linenum@out{%
1196       \string\@ref[\the\insert@count] []}}%
1197     \next%
1198     \ifnum\insert@count<1%
1199       \if@noneed@Footnote\else%

```



```

1200 \led@err@EdtextWithoutFootnote%
1201 \fi%
1202 \fi%
1203 \fi}%
1204
1205 %

```

**\startsub** \startsub and \endsub turn sub-lineation on and off, by writing appropriate instructions to the line-list file. When sub-lineation is in effect, the line number counter is frozen and the sub-line counter advances instead. If one of these commands appears in the middle of a line, it does not take effect until the next line; in other words, a line is counted as a line or sub-line depending on what it started out as, even if that changes in the middle.

We tinker with \lastskip because a command of either sort really needs to be attached to the last word preceding the change, not the first word that follows the change. This is because sub-lineation will often turn on and off in mid-line—stage directions, for example, often are mixed with dialogue in that way—and when a line is mixed we want to label it using the system that was in effect at its start. But when sub-lineation begins at the very start of a line we have a problem, if we don't put in this code.

```

1206
1207
1208 \newcommand*{\startsub}{\dimen0\lastskip
1209 \ifdim\dimen0>0pt \unskip \fi
1210 \ifledRcol \write\linenum@outR{\string\sub@on}%
1211 \else \write\linenum@out{\string\sub@on}%
1212 \fi
1213 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1214 \def\endsub{\dimen0\lastskip
1215 \ifdim\dimen0>0pt \unskip \fi
1216 \ifledRcol \write\linenum@outR{\string\sub@off}%
1217 \else \write\linenum@out{\string\sub@off}%
1218 \fi
1219 \ifdim\dimen0>0pt \hskip\dimen0 \fi}
1220
1221 %

```

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

```

1222 \newcommand*{\advanceline}[1]{\leavevmode%
1223 \ifledRcol \write\linenum@outR{\string\adv[#1]}%
1224 \else \write\linenum@out{\string\adv[#1]}%
1225 \fi%
1226 }
1227 %

```

**\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.



```

1228 \newcommand*{\setline}[1]{%
1229 \leavevmode%
1230 \ifnum#1<\z@
1231 \led@warn@BadSetline
1232 \else
1233 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1234 \else \write\linenum@out{\string\@set[#1]}%
1235 \fi
1236 \fi}
1237
1238
1239 %

```

**\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.

```

1240 \newcommand*{\setlinenum}[1]{%
1241 \ifnum#1<\z@
1242 \led@warn@BadSetlinenum
1243 \else
1244 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1245 \else \write\linenum@out{\string\l@d@set[#1]} \fi
1246 \fi}
1247
1248
1249 %

```

**\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.

```

1250 \newcommand*{\startlock}{%
1251 \ifledRcol \write\linenum@outR{\string\lock@on}%
1252 \else \write\linenum@out{\string\lock@on}%
1253 \fi}
1254 \def\endlock{%
1255 \ifledRcol \write\linenum@outR{\string\lock@off}%
1256 \else \write\linenum@out{\string\lock@off}%
1257 \fi}
1258
1259 %

```

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

```

\ifl@dskipversenumber
\l@dskipnumbertrue \newif\ifl@dskipnumber
\l@dskipnumberfalse \newif\ifl@dskipversenumber%
\skipnumbering \newcommand*{\skipnumbering}{%
\leavevmode%
\ifledRcol%

```



```

1265 \ifistanza%
1266     \write\linenum@outR{\string\n@num@stanza}%
1267 \else%
1268     \write\linenum@outR{\string\n@num}%
1269 \fi%
1270 \advanceline{-1}%
1271 \else%
1272     \ifistanza%
1273         \write\linenum@out{\string\n@num@stanza}%
1274     \else%
1275         \write\linenum@out{\string\n@num}%
1276     \fi%
1277     \advanceline{-1}%
1278 \fi%
1279 }%
1280
1281 %

```

## 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. 123). 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.

```
1282 \list@create{\end@lemmas}
1283 %
```

`\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.



```
1284 \newcommand{\dummy@edtext}[2]{#1}
1285 %
```

`\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`.

```
1286 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1287 %
```

We are going to need another macro that takes one argument and ignores it entirely. This is supplied by the  $\TeX$  `\@gobble{<arg>}`.

`\no@expands` We need to turn off macro expansion for certain sorts of macros we are likely to see  
`\morenoexpands` 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>26</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

<sup>26</sup>Since 'control sequences equivalent to characters are not expandable'—*The TeXbook*, answer to Exercise 20.14.



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<sub>TeX</sub> or Xe<sub>La</sub>TeX.)

```

1288 \newcommand*{\no@expands}{%
1289   \let\select@lemmafnt=0%
1290   \let\startsub=\relax \let\endsub=\relax
1291   \let\startlock=\relax \let\endlock=\relax
1292   \let\edlabel=\@gobble
1293   \let\setline=\@gobble \let\advanceline=\@gobble
1294   \let\sameword\sameword@inedtext%
1295   \let\edtext=dummy@edtext
1296   \l@dtabnoexpands
1297   \morenoexpands}
1298 \let\morenoexpands=\relax
1299
1300 %

```

**\@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.

```

1301 \newcommand{\@tag}{}
1302 %

```

**\@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 {\lemma} argument.

```

1303 \newcount\@edtext@level%
1304 \@edtext@level=0%
1305 %

```

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

```

1306 \newcommand{\edtext}[2]{\leavevmode%
1307 %

```

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

```

1308   \ifnumberedpar%
1309 %

```

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

```

1310   \global\advance\@edtext@level by 1%
1311 %

```

By default, we do not use \lemma

```

1312   \global\@lemmacommand@false%
1313 %

```



```

1314 \begingroup%
1315 %

```

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`.

```

1316 \ifledRcol%
1317 \ifcsundef{sw@list@edtextR@the\@edtext@level}%
1318 {\global\let\sw@inthisedtext\empty}%
1319 {\ifcseempty{sw@list@edtextR@the\@edtext@level}%
1320 {\global\let\sw@inthisedtext\empty}%
1321 {\expandafter\glp\csname sw@list@edtextR@the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1322 }%
1323 \else%
1324 \ifcsundef{sw@list@edtext@\the\@edtext@level}%
1325 {\global\let\sw@inthisedtext\empty}%
1326 {\ifcseempty{sw@list@edtext@\the\@edtext@level}%
1327 {\global\let\sw@inthisedtext\empty}%
1328 {\expandafter\glp\csname sw@list@edtext@\the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1329 }%
1330 \fi%
1331 %

```

**\@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.

```

1332 \global\renewcommand{\@tag}{%
1333 \no@expands #1%
1334 }%
1335 %

```

**\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`.

```

1336 \set@line%
1337 %

```

`\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`.

```

1338 \ifledRcol \global\insert@countR \z@%
1339 \else \global\insert@count \z@ \fi%
1340 %

```



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.

```
1341 \ignorespaces #2\relax%
1342 %
```

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

```
1343 \ifundefined{xpg@main@language}{%if not polyglossia
1344 \flag@start}%
1345 {\if@RTL\flag@end\else\flag@start\fi%
1346 }%
1347 %
```

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

```
1348 \if@lemmacommand%
1349 \ifledRcol%
1350 \write\linenum@outR{\string\@lemma}%
1351 \else%
1352 \write\linenum@out{\string\@lemma}%
1353 \fi%
1354 \fi%
1355 %
```

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.

```
1356 \endgroup%
1357 \showlemma{#1}%
1358 %
```

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.

```
1359 \ifx\end@lemmas\empty \else%
1360 \gl@p\end@lemmas\to\x@lemma%
1361 \x@lemma%
1362 \global\let\x@lemma=\relax%
1363 \fi%
1364 \ifundefined{xpg@main@language}{%if not polyglossia
1365 \flag@end}%

```



```

1366     {\if@RTL\flag@start\else\flag@end\fi% With polyglossia, you must
      track whether the language reads left to right (English) or right to left
      (Arabic).
1367     }%
1368 %

```

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`.

```

1369     \global\@noneed@Footnotefalse%
1370     \global\advance\@edtext@level by -1%
1371     \global\@lemmacommand@false%
1372 %

```

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

```

1373     \else%
1374     \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
      led@err@edtextoutsidestart%
1375     \fi%
1376 }%
1377
1378 \newcommand*{\flag@end}{%
1379     \ifledRcol%
1380         \write\linenum@outR{}}%
1381     \else%
1382         \write\linenum@out{}}%
1383     \fi}%
1384
1385 %

```

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

```

1386 \newif\ifnumberline
1387 \numberlinetrue
1388 %

```

`\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.

```

1389 \newcommand*{\set@line}{%
1390   \ifledRcol
1391     \ifx\line@listR\empty
1392       \global\noteschanged@true
1393       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1394     \else
1395       \glp\line@listR\to\@tempb
1396       \xdef\l@d@nums{\@tempb|\edfont@info}%
1397       \global\let\@tempb=undefined
1398     \fi
1399   \else
1400     \ifx\line@list\empty
1401       \global\noteschanged@true
1402       \xdef\l@d@nums{000|000|000|000|000|000|000|\edfont@info}%
1403     \else
1404       \glp\line@list\to\@tempb
1405       \xdef\l@d@nums{\@tempb|\edfont@info}%
1406       \global\let\@tempb=undefined
1407     \fi
1408   \fi}
1409
1410 %

```

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

```

1411 \newcommand*{\edfont@info}{\f@encoding/\f@family/\f@series/\f@shape}
1412
1413 %

```

## 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. 107).

```

1414 \unless\ifnocritical@
1415 \newcommand*{\lemma}[1]{%
1416   \global\@lemmacommand@true%
1417   \global\renewcommand{\@tag}{%
1418     \no@expands #1%
1419   }%
1420   \ignorespaces%
1421 }%
1422 %

```



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

```

1423 \newcommand{\@lemma}{%
1424   \booltrue{lemmacommand@the\edtext@level}%
1425 }%
1426 \fi
1427 %

```

**\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.

```

1428 \newif\if@lemmacommand@%
1429 %

```

### 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. 82): 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.

```

1430 \newcommand*{\linenum}[1]{%
1431   \xdef\@tempa{#1|}|}|}|}|}|noexpand\\l@d@nums}%
1432   \global\let\l@d@nums=\empty
1433   \expandafter\line@set\@tempa|\\ignorespaces}
1434 %

```

**\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.

```

1435 \def\line@set#1|#2|#3|#4|{%
1436   \gdef\@tempb{#1}%
1437   \ifx\@tempb\empty
1438     \l@d@add{#3}%
1439   \else
1440     \l@d@add{#1}%
1441   \fi
1442   \gdef\@tempb{#4}%
1443   \ifx\@tempb\empty\else

```



```

1444 \l@d@add{|\}\line@set#2\|#4\|}%
1445 \fi}
1446 %

```

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

```

1447 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1448
1449 %

```

## 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>27</sup>

Because there is a bug with `\detokenize` and  $\text{\LaTeX}$  when using non BMP characters<sup>28</sup>, we detokenize only for not  $\text{\LaTeX}$  engines. In any case, in  $\text{\LaTeX}$ , 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.

```

1450 \newcommand{\get@sw@txt}[1]{%
1451   \ifxetex%
1452     \xdef\sw@txt{#1}%
1453   \else%
1454     \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
1455   \fi%
1456 }%
1457 %

```

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

```

1458 \newcommand{\sameword}[2][1,usedefault]{%
1459   \leavevmode%
1460   \get@sw@txt{#2}%
1461 %

```

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

```

1462   \unless\ifledRcol%
1463     \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1464 %

```

Then, write its value to the numbered file.

<sup>27</sup>See <http://tex.stackexchange.com/q/244538/7712>.

<sup>28</sup><http://sourceforge.net/p/xetex/bugs/108/>



```

1465 \protected@write\linenum@out{}\string\sw{\sw@txt}{\csuse{sw@sw@txt
}}{#1}}%
1466 %

```

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

```

1467 \else%
1468 \csnumgdef{sw@sw@txt}{\csuse{sw@sw@txt}+1}%
1469 \protected@write\linenum@outR{}\string\sw{\sw@txt}{\csuse{sw@sw@txt
}}{#1}}%
1470 \fi%
1471 %

```

And print the word.

```

1472 #2%
1473 }%
1474 %

```

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

```

\if@addsw75 \newif\if@addsw%
1476 %

```

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

```

1477 \newcommand{\sw}[3]{%
1478 \get@sw@txt{#1}%
1479 \unless\ifledRcol%
1480 %

```

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

```

1481 \csxdef{sw@sw@txt @\the\absline@num @\the\section@num}{#2}%
1482 %

```

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

```

1483 \numdef{\prev@line}{\the\absline@num-1}%
1484 \ifcsundef{sw@sw@txt @\prev@line @\the\section@num}{%
1485 \csnumgdef{sw@sw@txt @\prev@line @\the\section@num}{#2-1}%
1486 }{}%
1487 %

```

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

```

1488 \numdef{\the@sw}{#2-\csuse{sw@sw@txt @\prev@line @\the\section@num}}%
1489 %

```

And do the same thing for the right side.



```

1490 \else%
1491 \csxdef{sw@sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1492 \numdef{\prev@line}{\the\absline@numR-1}%
1493 \ifcsundef{sw@sw@txt @\prev@line @\the\section@numR @R}{%
1494 \csnumgdef{sw@sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1495 }{}%
1496 \numdef{\the@sw}{#2-\csuse{sw@sw@txt @\prev@line @\the\section@numR @R
1497 }}%
1498 \fi%
1499 %

```

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

```

1499 \@tempcnta=\@edtext@level
1500 \@whilenum{\@tempcnta>0}\do{%
1501 \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1502 {%
1503 \addswfalse%
1504 \notbool{lemmacommand@\the\@tempcnta}%
1505 {\addswtrue}%
1506 {\IfStrEq{#3}{inlemma}%
1507 {\addswtrue}%
1508 {%
1509 \def\do##1{%
1510 \ifnumequal{##1}{\the\@tempcnta}%
1511 {\addswtrue\listbreak}%
1512 }%
1513 }%
1514 \docsvlist{#3}%
1515 }%
1516 }%
1517 \if@addsw%
1518 \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1519 \ifledRcol%
1520 \xright@appenditem{\the@sw}{\the\absline@numR}\to\@tmp%
1521 \else%
1522 \xright@appenditem{\the@sw}{\the\absline@num}\to\@tmp%
1523 \fi%
1524 \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1525 \fi%
1526 }%
1527 }%
1528 \advance\@tempcnta by -1%
1529 }%
1530 }%
1531 %

```

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

```

1532 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1533 \get@sw@txt{#2}%

```



```

1534 \unless\ifledRcol@%
1535 %

```

Just a precaution.

```

1536 \ifx\sw@list@inedtext\empty%
1537 \def\the@sw{999}%
1538 \def\this@absline{-99}%
1539 \else%
1540 %

```

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

```

1541 \gl@p\sw@list@inedtext\to\@tmp%
1542 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1543 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1544 \fi%
1545 %

```

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

```

1546 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@num}{%
1547 \numdef{\prev@line}{\this@absline-1}%
1548 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@num}-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1549 }%
1550 {\numdef{\sw@atthisline}{0}}%
1551 %

```

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

```

1552 \ifnumgreater{\sw@atthisline}{1}%
1553 {\showwordrank{#2}{\the@sw}}%
1554 {#2}%
1555 %

```

And the same for right side.

```

1556 \else%
1557 \ifx\sw@list@inedtext\empty%
1558 \def\the@sw{999}%
1559 \def\this@absline{-99}%
1560 \else%
1561 \gl@p\sw@list@inedtext\to\@tmp%
1562 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1563 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1564 \fi%
1565 \ifcsdef{sw@\sw@txt @\this@absline @\the\section@numR @R}{%
1566 \numdef{\prev@line}{\this@absline-1}%
1567 \numdef{\sw@atthisline}{\csuse{sw@\sw@txt @\this@absline @\the\
section@numR @R}-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1568 }%

```



```

1569     {\numdef{\sw@atthisline}{0}}%
1570     \ifnumgreater{\sw@atthisline}{1}%
1571         {\showwordrank{#2}{\the@sw}}%
1572         {#2}%
1573     \fi%
1574 }%
1575 %

```

```

\showwordrank % Finally, the way the rank will be printed.
1577 \newcommand{\showwordrank}[2]{%
1578     #1\textsuperscript{#2}%
1579 }%
1580 %

```

## 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.

```

1581 \newbox\raw@text
1582 \newif\ifnumberedpar@
1583 \newcount\num@lines
1584 \newbox\one@line
1585 \newcount\par@line
1586 %

```

`\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.

```

1587
1588 \newcommand{\AtEveryPstart}[1]{%
1589   \ifstrempy{#1}%
1590     {\xdef\at@every@pstart{}}%
1591     {\gdef\at@every@pstart{\noindent#1}}%
1592 }%
1593 \xdef\at@every@pstart{}%
1594
1595 \newcounter{pstart}
1596 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
1597 \newif\ifnumberpstart
1598 \numberpstartfalse
1599 \newif\iflabelpstart
1600 \labelpstartfalse
1601 \newcommandx*{\pstart}[1][1]{%
1602   \normal@pars%
1603   \ifstrempy{#1}{\at@every@pstart}{\noindent#1}%
1604   \ifautopar%
1605     \autopar%
1606   \fi%
1607   \ifluatex%
1608     \edef\l@luatextextdir@L{\the\textdir}%
1609   \fi%
1610   \if@nobreak%
1611     \let\@oldnobreak\@nobreaktrue%
1612   \else%
1613     \let\@oldnobreak\@nobreakfalse%
1614   \fi%
1615   \@nobreaktrue%
1616   \ifnumbering \else%
1617     \led@err@PstartNotNumbered%
1618     \beginnumbering%
1619   \fi%
1620   \ifnumberedpar@%
1621     \led@err@PstartInPstart%
1622     \pend%
1623   \fi%
1624   \list@clear{\inserts@list}%
1625   \global\let\next@insert=\empty%
1626   \begingroup\normal@pars%
1627   \global\advance \l@dnumpstartsL\@ne
1628   \global\setbox\raw@text=\vbox\bgroup%
1629     \ifautopar\else%
1630     \ifnumberpstart%
1631       \ifinstanza\else%
1632       \ifsidepstartnum\else%
```



```

1633     \thepstart%
1634     \fi%
1635     \fi%
1636     \fi%
1637     \fi%
1638     \numberedpar@true%
1639     \iflabelpstart\protected@edef\@currentlabel%
1640         {\p@pstart\thepstart}
1641     \fi%
1642     \l@dzeropenalties%
1643     \ignorespaces%because not automatically ignored if an optional argument
is used (classical TeX behavior for space after commands)
1644 }
1645 %

```

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

```

1646 \newcommandx*{\pend}[1][1]{\ifnumbering \else%
1647     \led@err@PendNotNumbered%
1648     \fi%
1649     \global\l@dskipversenumberfalse%
1650     \ifnumberedpar@ \else%
1651         \led@err@PendNoPstart%
1652     \fi%
1653 %

```

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.

```

1654     \l@dzeropenalties%
1655     \endgraf\global\num@lines=\prevgraf\egroup%
1656     \global\par@line=0%
1657 %

```

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`.

```

1658     \csnumdef{pstartline}{0}%
1659     \loop\ifvbox\raw@text%
1660         \csnumdef{pstartline}{\pstartline+1}%
1661         \do@line%
1662         \ifbypstart@%
1663             \ifnumequal{\pstartline}{1}{%
1664                 \bgroup%
1665                 \let\leavevmode\relax%

```



```

1666     \setline{1}%
1667     \egroup%
1668     \resetprevline@{}%
1669     \fi%
1670     \repeat%
1671 %

```

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

```

1672     \flush@notes%
1673     \endgroup%
1674     \ignorespaces%
1675 %

```

Increase pstart counter.

```

1676     \ifnumberpstart%
1677         \pstartnumtrue%
1678     \fi%
1679     \addtocounter{pstart}{1}%
1680 %

```

Restore paragraph, nobreak setting and autopar setting.

```

1681     \normal@pars%
1682     \@oldnobreak%
1683     \ifautopar%
1684         \autopar%
1685     \fi%
1686 %

```

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

```

1687     \ifstrempy{#1}{\at@every@pend}{\noindent#1}%
1688 }
1689 %
1690 %

```

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.

```

\AtEveryPend91
\at@every@pend92 \newcommand{\AtEveryPend}[1]{%
1693     \ifstrempy{#1}%
1694         {\xdef\at@every@pend{}}%
1695         {\gdef\at@every@pend{\noindent#1}}%
1696     }%
1697     \xdef\at@every@pend{}%
1698 %
1699 %

```

**\l@dzero penalties** A macro to zero penalties for \pend or \pstart.



```

1700 \newcommand*{\l@dzzeropenalties}{%
1701   \brokenpenalty \z@ \clubpenalty \z@
1702   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1703   \postdisplaypenalty \z@ \widowpenalty \z@}
1704
1705 %

```

`\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.

```

1706 \newif\ifautopar
1707 \autoparfalse
1708 \newcommand*{\autopar}{
1709   \ifledRcol
1710     \ifnumberingR \else
1711       \led@err@AutoparNotNumbered
1712     \beginnumberingR
1713     \fi
1714   \else
1715     \ifnumbering \else
1716       \led@err@AutoparNotNumbered
1717     \beginnumbering
1718     \fi
1719   \fi
1720   \autopartrue
1721   \everypar{\setbox0=\lastbox
1722     \endgraf \vskip-\parskip

```



```

1723 \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\
fi\fi
1724 \let\par=\pend}%
1725 \ignorespaces}
1726 %

```

**\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.

```

1727 \newcommand*{\normal@pars}{\everypar{}\let\par\endgraf}
1728
1729 %

```

**\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.

```

1730 \newif\ifautopar@pause
1731 %

```

## 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**

```

1732 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1733 \newcommand*{\do@line}{%
1734   {\vbadness=10000
1735     \splittopskip=\z@
1736     \do@linehook
1737   \l@demptyd@ta
1738     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1739   \unvbox\one@line \global\setbox\one@line=\lastbox
1740   \getline@num
1741   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{\}
1742   \ifnum\@lock>\@one
1743     \inserthangingsymboltrue
1744   \else
1745     \inserthangingsymbolfalse
1746   \fi
1747   \check@pb@in@verse
1748   \ifl@dhidenumber%
1749     \global\l@dhidenumberfalse%
1750     \f@x@l@cks%
1751   \else%
1752     \affixline@num%
1753   \fi%
1754   %

```



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

```

1755 \xifinlist{\the\l@dumpstartsL}{\eled@sections@@}%
1756     {\print@eledsection}%
1757     {\print@line}%
1758 \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{\}
1759 }%
1760 %

```

### VII.2.2 Process for “normal” line

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

```

1761 \def\print@line{
1762 %

```

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

```

1763     \affixpstart@num%
1764 %

```

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

```

1765     \hb@xt@ \linewidth{\%
1766 %

```

User hook.

```

1767     \do@insidelinehook%
1768 %

```

Left line number

```

1769     \l@dld@ta%
1770 %

```

Restore marginal and footnotes.

```

1771     \add@inserts\affixside@note%
1772 %

```

Print left notes.

```

1773     \l@dlsn@te
1774 %

```

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

```

1775     {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1776 %

```

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

```

1777     \ifluatex%
1778     \textdir\l@luatextextdir@L%
1779     \fi%
1780 %

```



Insert, if needed, the hanging symbol.

```
1781 \inserthangingsymbol %Space kept for backward compatibility
1782 %
```

And so, print the line.

```
1783 \l@dunhbox@line{\one@line}}%
1784 %
```

Right line number

```
1785 \ledrlfill\l@drd@ta%
1786 %
```

Print right notes.

```
1787 \l@drsn@te
1788 }}%
1789 %
```

And reinsert penalties (for page breaking)...

```
1790 \add@penalties%
1791 }
1792 %
```

### 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.

```
1793 \def\print@eledsection{%
1794 \add@inserts\affixside@note%
1795 \numdef{\temp@}{\l@dnumpstartsL-1}%
1796 \xifinlist{\temp@}{\eled@sections@@}{\@nbreaktrue}{\@nbreakfalse}%
1797 \@eled@sectioningtrue%
1798 \csuse{eled@sectioning@the\l@dnumpstartsL}%
1799 \@eled@sectioningfalse%
1800 \global\csundef{eled@sectioning@the\l@dnumpstartsL}%
1801 \if@RTL%
1802 \hspace{-3\paperwidth}%
1803 {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1804 \else%
1805 \hspace{3\paperwidth}%
1806 {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1807 \fi%
1808 \vskip-\baselineskip%
1809 }
1810 %
```



### VII.2.4 Hooks

`\do@linehook` Two hooks into `\do@line`. The first is called at the beginning of `\do@line`, the second  
`\do@insidelinehook` is called in the line box. The second can, for example, have a `\markboth` command  
inside, the first can not.

```
1811 \newcommand*\do@linehook{}{}
1812 \newcommand*\do@insidelinehook{}{}
1813 %
```

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

```
1814 \newcommand*\dolinehook[1]{\gdef\do@linehook{#1}}%
1815 \newcommand*\doinsidelinehook[1]{\gdef\do@insidelinehook{#1}}%
1816
1817 %
```

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

`\l@demptyd@ta` Nulls the `\l@demptyd@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  
`\l@drd@ta` notes.

```
1818 \l@dcsnotetext \newcommand*\l@demptyd@ta{}{}%
1819 \l@dcsnotetext@l \gdef\l@dld@ta{}{}%
1820 \l@dcsnotetext@r \gdef\l@drd@ta{}{}%
1821 \gdef\l@dcsnotetext@l{}{}%
1822 \gdef\l@dcsnotetext@r{}{}%
1823 \gdef\l@dcsnotetext{}{}
1824
1825 %
```

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

```
1826 \l@dlsn@te \newcommand*\l@dlsn@te{}{}%
1827 \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1828 \l@dlsn@te \newcommand*\l@dlsn@te{}{}%
1829 \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1830
1831 %
```

`\ledllfill` These macros are called at the left (`\ledllfill`) and the right (`\ledllfill`) of each  
`\ledrlfill` numbered line. The initial definitions correspond to the original code for `\do@line`.

```
1832 \newcommand*\ledllfill{}{\hfil}
1833 \newcommand*\ledrlfill{}{}
1834
1835 %
```



## 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.

```

1836 \newcommand*{\getline@num}{%
1837   \global\advance\absline@num \@ne%
1838   \do@actions
1839   \do@ballast
1840   \ifnumberline
1841     \ifsublines@
1842       \ifnum\sub@lock<\tw@
1843         \global\advance\subline@num \@ne
1844       \fi
1845     \else
1846       \ifnum\@lock<\tw@
1847         \global\advance\line@num \@ne
1848         \global\subline@num \z@
1849       \fi
1850     \fi
1851   \fi
1852 }
1853 %

```

`\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. 136).

`\ballast@count` First we set up the required counters; they are initially set to zero, and will remain so unless you type `\setcounter{ballast}{<some figure>}` in your document.

`\c@ballast`

```

1854 \newcount\ballast@count
1855 \newcounter{ballast}
1856 \setcounter{ballast}{0}
1857 %

```

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.

```

1858 \newcommand*{\do@ballast}{\global\ballast@count \z@
1859   \begingroup
1860     \advance\absline@num \@ne
1861     \ifnum\next@actionline=\absline@num
1862       \ifnum\next@action>-1001\relax
1863         \global\advance\ballast@count by -\c@ballast
1864       \fi
1865     \fi

```



```

1866 \endgroup}
1867 %

```

`\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.

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`.

```

1868 \newcommand*{\do@actions}{%
1869   \global\let\do@actions@next=\relax
1870   \ifnum\absline@num<\next@actionline\else
1871 %

```

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.

```

1872   \ifnum\next@action>-1001
1873     \global\page@num=\next@action
1874     \ifbypage@
1875       \global\line@num=\z@ \global\subline@num=\z@
1876       \resetprevline@
1877   \fi
1878 %

```

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`.)

```

1879   \else
1880     \ifnum\next@action<-4999
1881       \@l@dttempcnta=-\next@action
1882       \advance\@l@dttempcnta by -5001
1883       \ifsublines@
1884         \global\subline@num=\@l@dttempcnta
1885       \else
1886         \global\line@num=\@l@dttempcnta
1887     \fi
1888 %

```

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

```

1889   \else
1890     \@l@dttempcnta=-\next@action
1891     \advance\@l@dttempcnta by -1000
1892     \do@actions@fixedcode
1893   \fi
1894 \fi
1895 %

```



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.

```

1896   \ifx\actionlines@list\empty
1897       \gdef\next@actionline{1000000}%
1898   \else
1899       \gl@p\actionlines@list\to\next@actionline
1900       \gl@p\actions@list\to\next@action
1901       \global\let\do@actions@next=\do@actions
1902   \fi
1903 \fi
1904 %

```

Make the recursive call, if necessary.

```

1905 \do@actions@next}
1906
1907 %

```

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

```

1908 \newcommand*{\do@actions@fixedcode}{%
1909   \ifcase\@l@dttempcnta
1910   \or% % 1001
1911       \global\sublines@true
1912   \or% % 1002
1913       \global\sublines@false
1914   \or% % 1003
1915       \global\@lock=\@ne
1916   \or% % 1004
1917       \ifnum\@lock=\tw@
1918           \global\@lock=\thr@@
1919       \else
1920           \global\@lock=\z@
1921       \fi
1922   \or% % 1005
1923       \global\sub@lock=\@ne
1924   \or% % 1006
1925       \ifnum\sub@lock=\tw@
1926           \global\sub@lock=\thr@@
1927       \else
1928           \global\sub@lock=\z@
1929       \fi
1930   \or% % 1007
1931       \l@dskipnumbertrue
1932   \or% % 1008
1933       \l@dskipversenumbertrue%
1934   \or% % 1009
1935       \l@dhiddenumbertrue

```



```

1936 \else
1937 \led@warn@BadAction
1938 \fi}
1939
1940
1941 %

```

## 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.

```

1942 \newcommand*{\affixline@num}{\%
1943 %

```

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).

```

1944 \ifledgroupnotesL\else
1945 \ifnumberline
1946 \ifl@dskipnumber
1947 \global\l@dskipnumberfalse
1948 \else
1949 \ifsublines@
1950 \@l@tempcntb=\subline@num
1951 \ifnum\subline@num>\c@firstsublinenum
1952 \@l@tempcnta=\subline@num
1953 \advance\@l@tempcnta by-\c@firstsublinenum
1954 \divide\@l@tempcnta by\c@sublinenumincrement
1955 \multiply\@l@tempcnta by\c@sublinenumincrement
1956 \advance\@l@tempcnta by\c@firstsublinenum
1957 \else
1958 \@l@tempcnta=\c@firstsublinenum
1959 \fi
1960 %

```



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.

```
1961      \ch@cksub@l@ck
1962 %
```

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

```
1963      \else
1964      \@l@tempcntb=\line@num
1965 %
```

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

```
1966      \ifx\linenumberlist\empty
1967      \ifnum\line@num>\c@firstlinenum
1968      \@l@tempcnta=\line@num
1969      \advance\@l@tempcnta by-\c@firstlinenum
1970      \divide\@l@tempcnta by\c@linenumincrement
1971      \multiply\@l@tempcnta by\c@linenumincrement
1972      \advance\@l@tempcnta by\c@firstlinenum
1973      \else
1974      \@l@tempcnta=\c@firstlinenum
1975      \fi
1976      \else
1977 %
```

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

```
1978      \@l@tempcnta=\line@num
1979      \edef\rem@inder{\,\linenumberlist,\number\line@num,}%
1980      \edef\sc@n@list{\def\noexpand\sc@n@list
1981      ###1,\number\@l@tempcnta,###2|\def\noexpand\rem@inder
1982      {####2}}}%
1983      \sc@n@list\expandafter\sc@n@list\rem@inder|
1984      \ifx\rem@inder\empty%
1985      \advance\@l@tempcnta\@ne
1986      \fi
1987 %
```

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

```
1988      \ch@ck@l@ck
1989      \fi
1990 %
```

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

```
1991      \ifnum\@l@tempcnta=\@l@tempcntb
1992      \ifl@dskipversenumber\else
1993 %
```



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`

```

1994         \if@twocolumn
1995             \if@firstcolumn
1996                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1997             \else
1998                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1999             \fi
2000         \else
2001             \@l@tempcntb=\line@margin
2002             \ifnum\@l@tempcntb>\@ne
2003                 \advance\@l@tempcntb \page@num
2004             \fi
2005             \ifodd\@l@tempcntb
2006                 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
2007             \else
2008                 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
2009             \fi
2010         \fi
2011     \fi
2012 \fi
2013 %

```

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

```

2014         \f@x@l@cks
2015     \fi
2016 \fi
2017 \fi
2018 }
2019 %
2020 %

```

`\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.



```

2021 \newcommand*{\ch@cksub@l@ck}{%
2022   \ifcase\sub@lock
2023   \or
2024     \ifnum\sublock@disp=\@ne
2025       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2026     \fi
2027   \or
2028     \ifnum\sublock@disp=\tw@ \else
2029       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2030     \fi
2031   \or
2032     \ifnum\sublock@disp=\z@
2033       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2034     \fi
2035   \fi}
2036 %

```

Similarly for line numbers.

```

2037 \newcommand*{\ch@ck@l@ck}{%
2038   \ifcase\@lock
2039   \or
2040     \ifnum\lock@disp=\@ne
2041       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2042     \fi
2043   \or
2044     \ifnum\lock@disp=\tw@ \else
2045       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2046     \fi
2047   \or
2048     \ifnum\lock@disp=\z@
2049       \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2050     \fi
2051   \fi}
2052 %

```

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

```

2053 \newcommand*{\f@x@l@cks}{%
2054   \ifcase\@lock
2055   \or
2056     \global\@lock=\tw@
2057   \or \or
2058     \global\@lock=\z@
2059   \fi
2060   \ifcase\sub@lock
2061   \or
2062     \global\sub@lock=\tw@
2063   \or \or
2064     \global\sub@lock=\z@
2065   \fi}

```



2066  
2067 %

## 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 2068
\rightpstartnum 2069 \newif\ifsidepstartnum
\ifsidepstartnum 2070 \newcommand*{\affixpstart@num}{%
2071   \ifsidepstartnum
2072     \if@twocolumn
2073       \if@firstcolumn
2074         \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2075       \else
2076         \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2077       \fi
2078     \else
2079       \@l@tempcntb=\line@margin
2080       \ifnum\@l@tempcntb>\@ne
2081         \advance\@l@tempcntb \page@num
2082       \fi
2083       \ifodd\@l@tempcntb
2084         \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2085       \else
2086         \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2087       \fi
2088     \fi
2089   \fi
2090 }
2091 %
2092
2093
2094 \newif\ifpstartnum
2095 \pstartnumtrue
2096 \newcommand*{\leftpstartnum}{
2097   \ifpstartnum\thepstart
2098   \kern\linenumsep\fi

```



```

2099 \global\pstartnumfalse
2100 }
2101 \newcommand*{\rightpstartnum}{
2102   \ifpstartnum
2103   \kern\linenumsep
2104   \thepstart
2105   \fi
2106   \global\pstartnumfalse
2107 }
2108 %

```

## 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.

```

2109 \list@create{\inserts@list}
2110 %

```

`\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`.

```

2111 \newcommand*{\add@inserts}{%
2112   \global\let\add@inserts@next=\relax
2113   %

```

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

```

2114   \ifx\inserts@list\empty \else
2115   %

```



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.

```

2116 \ifx\next@insert\empty
2117   \ifx\insertlines@list\empty
2118     \global\noteschanged@true
2119     \gdef\next@insert{100000}%
2120   \else
2121     \gl@p\insertlines@list\to\next@insert
2122   \fi
2123 \fi
2124 %

```

If the next insert's for this line, tack it on (and then erase the contents of the insert macro, as it could be quite large). In that case, we also set `\add@inserts@next` so that we will call ourselves recursively: there might be another insert for this same line.

```

2125 \ifnum\next@insert=\absline@num
2126   \gl@p\inserts@list\to\@insert
2127   \@insert
2128   \global\let\@insert=\undefined
2129   \global\let\next@insert=\empty
2130   \global\let\add@inserts@next=\add@inserts
2131 \fi
2132 \fi
2133 %

```

Make the recursive call, if necessary.

```

2134 \add@inserts@next}
2135
2136 %

```

## 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. 127). Finally, the penalty is checked to see that it does not go below  $-10000$ .

```

2137 \newcommand*{\add@penalties}{\@l@dttempcnta=\ballast@count
2138   \ifnum\num@lines>\@ne
2139     \global\advance\par@line \@ne
2140   \ifnum\par@line=\@ne

```



```

2141 \advance\@l@dttempcnta \clubpenalty
2142 \fi
2143 \@l@dttempcntb=\par@line \advance\@l@dttempcntb \@ne
2144 \ifnum\@l@dttempcntb=\num@lines
2145 \advance\@l@dttempcnta \widowpenalty
2146 \fi
2147 \ifnum\par@line<\num@lines
2148 \advance\@l@dttempcnta \interlinepenalty
2149 \fi
2150 \fi
2151 \ifnum\@l@dttempcnta=\z@
2152 \relax
2153 \else
2154 \ifnum\@l@dttempcnta>-10000
2155 \penalty\@l@dttempcnta
2156 \else
2157 \penalty -10000
2158 \fi
2159 \fi}
2160
2161 %

```

### 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.

```

2162 \newcommand*{\flush@notes}{%
2163 \@xloop
2164 \ifx\inserts@list\empty \else
2165 \glp\inserts@list\to\@insert
2166 \@insert
2167 \global\let\@insert=\undefined
2168 \repeat}
2169
2170 %

```

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

```
2171 \def\@xloop#1\repeat{%
2172   \def\body{#1\expandafter\body\fi}%
2173   \body}
2174
2175 %
```

## XII Critical footnotes

The footnote macros are adapted from those in PLAIN T<sub>E</sub>X, but they differ in these respects: the outer-level commands must add other commands to a list macro rather than doing insertions immediately; there are many separate levels of the footnotes, not just one; and there are options to reformat footnotes into paragraphs or into multiple columns.

### XII.1 Fonts

Before getting into the details of formatting the notes, we set up some font macros. It is the notes that present the greatest challenge for our font-handling mechanism, because we need to be able to take fragments of our main text and print them in different forms: it is common to reduce the size, for example, without otherwise changing the fonts used.

`\select@lemmafont` `\select@lemmafont` is provided to set the right font for the lemma in a note. This macro extracts the font specifier from the line and page number cluster, and issues the associated font-changing command, so that the lemma is printed in its original font.

```
2176 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@@lemmafont#7|}
2177 \def\select@@lemmafont#1/#2/#3/#4|{%
2178   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2179   \selectfont}
2180
2181 %
```

### 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.

```
2182 \newcommand*{\footnoteoptions@}[3]{%
2183   \def\do##1{%
2184     \ifstrequal{#1}{L}{% In Leftside
2185       \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
2186         unexpanded{##1}}}{\to\inserts@list%
2187       \global\advance\insert@count \@ne% Increment the left insert
2188       counter.
2189     }%
2190   }%
2191 }
```



```

2188     {%
2189         \xright@appenditem{\noexpand\setkeys[mac]{#3footnoteoption}{\
unexpanded{##1}}}\to\inserts@listR%
2190         \global\advance\insert@countR \@ne% Increment the right insert
counter insert.
2191     }%
2192 }%
2193 \notblank{#2}{\docsvlist{#2}}{}% Parsing all options
2194 }
2195 %

```

### XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua<sup>®</sup>TeX is used.

```

2196 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2197     \ifstrequal{#1}{L}{%
2198         \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}\to\
inserts@list%Know the dir of lemma
2199         \global\advance\insert@count \@ne%
2200         \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}\to\
inserts@list%Know the dir of lemma
2201         \global\advance\insert@count \@ne%
2202     }%
2203     {%
2204         \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\textdir}}\to\
inserts@listR%Know the dir of lemma
2205         \global\advance\insert@countR \@ne%
2206         \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\pardir}}\to\
inserts@listR%Know the dir of lemma
2207         \global\advance\insert@countR \@ne%
2208     }%
2209 }
2210 %

```

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

```

2211 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2212     \ifstrequal{#1}{L}{%
2213         \if@RTL%
2214             \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}\to\
inserts@list%Know the language used in the lemma
2215             \global\advance\insert@count \@ne%
2216         \else
2217             \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}\to\
inserts@list%Know the language of lemma
2218             \global\advance\insert@count \@ne%

```



```

2219 \fi%
2220 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@list%Know the language of lemma
2221 \global\advance\insert@count \@ne%
2222 }%
2223 {%
2224 \if@RTL
2225 \xright@appenditem{{\csxdef{footnote@dir}{@RTLtrue}}}\to\
inserts@listR%Know the language of lemma
2226 \global\advance\insert@countR \@ne%
2227 \else
2228 \xright@appenditem{{\csxdef{footnote@dir}{@RTLfalse}}}\to\
inserts@listR%Know the language of lemma
2229 \global\advance\insert@countR \@ne%
2230 \fi
2231 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@listR%Know the language of lemma
2232 \global\advance\insert@countR \@ne%
2233 }%
2234 }
2235 %

```

## 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  $\TeX$  `\floatingpenalty`.

```

2236 \newcommand*{\footplitskips}{%
2237   \interlinepenalty=\interfootnotelinepenalty
2238   \unless\ifl@dprintingpages%
2239     \floatingpenalty=\@MM%
2240   \fi%
2241   \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2242   \leftskip=\z@skip \rightskip=\z@skip}
2243
2244 %

```

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

```

2245 \let\normalfootnoterule=\footnoterule
2246 %

```

## XII.6 Footnotes arrangement

### XII.6.1 User level macro

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

```

2247 \newcommandx{\Xarrangement}[2][1,usedefault]{%
2248   \def\do##1{%
2249     \csname Xarrangement@#2\endcsname{##1}%
2250   }%
2251   \ifstrepty{#1}%
2252     {%
2253       \dolistloop{\@series}%
2254     }%
2255   {
2256     \docsvlist{#1}%
2257   }%
2258 }%
2259 %

```



### 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\vAfootnote=\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.

```
2260 \newcommand*{\Xarrangement@normal}[1]{%
2261   \csgdef{series@display#1}{normal}
2262   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2263   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2264   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
2265   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2266   \expandafter\let\csname #1footnoterule\endcsname=%
2267                                     \normalfootnoterule
2268   \count\csname #1footins\endcsname=1000
2269   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2270   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2271   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2272   %
```

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.

```
2273   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
2274   side only
2275   %
```

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).

```
2275   \ifnoledgroup@else%
2276     \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2277     \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2278     \count\csname mp#1footins\endcsname=1000
```



```

2279 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2280 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2281 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2282 \fi
2283 }
2284
2285 %

```

**\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.

```

2286 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
2287 \insert\csname #1footins\endcsname\bgroup
2288 \noindent\csuse{Xhooknote@#1}%
2289 \csuse{Xnotefontsize@#1}%
2290 \footsplitskips
2291 \ifl@dpairing\ifl@dpadding\else%
2292 \setXnoteswidthliketwocolumns@{#1}%
2293 \fi\fi%
2294 \setXnotespositionliketwocolumns@{#1}%
2295 \spaceskip=\z@skip \xspaceskip=\z@skip
2296 \csname #1footfmt\endcsname #2{#1}\egroup}
2297 %

```

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

```

2298 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2299 \global\setbox\@nameuse{mp#1footins}\vbox{%
2300 \unvbox\@nameuse{mp#1footins}
2301 \noindent\csuse{Xhooknote@#1}%
2302 \csuse{Xnotefontsize@#1}%
2303 \hsize\columnwidth
2304 \@parboxrestore
2305 \color@begingroup
2306 \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2307
2308 %

```

**\normalfootfmt** **\normalfootfmt** is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 82), 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.

```

2309

```



```

2310 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2311 \Xledsetnormalparstuff{#4}%
2312 \hangindent=\csuse{Xhangindent@#4}
2313 \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2314 \strut{\printlinefootnote{#1}{#4}}%
2315 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2316 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
2317 Xlemmaseparator@#4}%
2318 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2319 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
2320 #4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2321 }}%
2322 #3\strut\par}
2323 %

```

**\normalfootstart** \normalfootstart is a standard footnote-starting macro, called in the output routine whenever there are footnotes of this series to be printed: it skips a bit and then draws a rule.

Any \footstart macro must put onto the page something that takes up space exactly equal to the \skipXfootins value for the associated series of notes. TeX makes page computations based on that \skip value, and the output pages will suffer from spacing problems if what you add takes up a different amount of space.

But if the skip \preXnotes@ is greater than 0 pt, it is used instead of \skipXfootins 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.

```

2323 \newcommand*{\normalfootstart}[1]{%
2324 %

```

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. 186. Here is part of this algorithm, when the block of notes are ready to be printed.

```

2325 \ifdimequal{0pt}{\preXnotes@}{}%
2326 {%
2327 \iftoggle{preXnotes@}{%
2328 \togglefalse{preXnotes@}%
2329 \skip\csname #1footins\endcsname=%
2330 \dimexpr\csuse{preXnotes@}+\csuse{Xafterterrule@#1}\relax%
2331 }%
2332 }%

```



```

2333 }%
2334 \vskip\skip\csize #1footins\endcsize%
2335 %

```

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. 184 for the general description of the problem.

```

2336 \leftskipOpt \rightskipOpt
2337 \ifl@dpairing\else%
2338   \hsize=\old@hsize%
2339 \fi%
2340 \setXnoteswidthliketwocolumns@{#1}%
2341 \setXnotespositionliketwocolumns@{#1}%
2342 %

```

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

```

2343 \print@Xfootnoterule{#1}%
2344 \noindent\leavevmode}
2345 %

```

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

```

2346 \newcommand*{\normalfootgroup}[1]{%
2347   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2348   \unvbox\csize #1footins\endcsize%
2349   \hsize=\old@hsize%
2350 }%
2351 %
2352 %

```

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

```

2353 \unless\ifnoledgroup@
2354 \newcommand*{\mpnormalfootgroup}[1]{%
2355   \vskip\skip\@nameuse{mp#1footins}
2356   \ifl@dpairing\ifparledgroup%
2357     \leavevmode\marks\parledgroup@{begin}%
2358     \marks\parledgroup@series{#1}%
2359     \marks\parledgroup@type{Xfootnote}%
2360   \fi\fi\normalcolor%
2361   \ifparledgroup%
2362     \ifl@dpairing%
2363     \else%
2364       \setXnoteswidthliketwocolumns@{#1}%
2365       \setXnotespositionliketwocolumns@{#1}%
2366       \print@Xfootnoterule{#1}%
2367     \fi%

```



```

2368 \else%
2369 \setXnoteswidthliketwocolumns@{#1}%
2370 \setXnotespositionliketwocolumns@{#1}%
2371 \printXfootnoterule{#1}%
2372 \fi%
2373 \setlength{\parindent}{0pt}
2374 {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2375 \unvbox\csname mp#1footins\endcsname}}
2376 \fi
2377 %

```

### 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.

```

2378 \newcommand*{\Xarrangement@paragraph}[1]{%
2379 \csgdef{series@display#1}{paragraph}
2380 \expandafter\newcount\csname #1prevpage@num\endcsname
2381 \expandafter\let\csname #1footstart\endcsname=\parafootstart
2382 \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2383 \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2384 \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2385 \count\csname #1footins\endcsname=1000
2386 \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2387 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2388 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2389 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2390 \para@footsetup{#1}
2391 %

```

And the extra setup for minipages.

```

2392 \ifnoledgroup@else
2393 \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2394 \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2395 \count\csname mp#1footins\endcsname=1000
2396 \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}

```



```

2397 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2398 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2399 \fi
2400 }
2401 %

```

**\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.

```

2402 \providecommand{\footfudgefiddle}{64}
2403 %

```

**\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 9pt) 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`.

```

2404 \newcommand*{\para@footsetup}[1]{\csuse{Xnotefontsize@#1}
2405 \setXnoteswidthliketwocolumns@{#1}%
2406 \dimen0=\baselineskip
2407 \multiply\dimen0 by 1024
2408 \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2409 \csxdef{#1footfudgefactor}{%
2410 \expandafter\strip@pt\dimen0 }}
2411
2412 %

```

`\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.

```

2413 \newcommand*{\parafootstart}[1]{%
2414 \rightskip=0pt \leftskip=0pt%
2415 \nottoggle{Xparindent@#1}{\parindent=\z@}{}%
2416 \ifdimequal{0pt}{\preXnotes@}{}%
2417 {%
2418 \iftoggle{preXnotes@}{%
2419 \togglefalse{preXnotes@}%
2420 \skip\csname #1footins\endcsname=%

```



```

2421 \dimexpr\csuse{preXnotes@}+\csuse{Xafterrule@#1}\relax%
2422 }%
2423 {}%
2424 }%
2425 \vskip\skip\csname #1footins\endcsname%
2426 \setXnoteswidthliketwocolumns@{#1}%
2427 \setXnotespositionliketwocolumns@{#1}%
2428 \print@Xfootnoterule{#1}%
2429 \let\bidirectional\everypar\@empty%
2430 \noindent\leavevmode}
2431 %

```

`\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>29</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>30</sup> So changes from one language to another will not invoke the proper hyphenation rules in such footnotes. Since critical editions often do deal with several languages, especially in a footnotes, we really ought to get this bit of code right.

To get around these problems, Wayne suggested emendations to the *TeXbook* versions of these macros which are broadly the same as those described by Michael: the central idea (also suggested by Donald Knuth in a letter to Michael) is to avoid collecting the text in an `\hbox` in the first place, but instead to collect it in a `\vbox` whose width is (virtually) infinite. The text is therefore typeset in unrestricted horizontal mode, as a paragraph consisting of a single long line. Later, there is an extra level of unboxing to be done: we have to unpack the `\vbox`, as well as the hboxes inside it, but that is not too hard. For details, we refer you to Michael's article, where the issues are clearly explained.<sup>31</sup> Michael's unboxing macro is called `\Xunvxh`: unvbox, extract the last line, and unhbox it.

<sup>29</sup>Michael Downes, 'Line Breaking in `\unhboxed` Text', *TUGboat* **11** (1990), pp. 605–612.

<sup>30</sup>See *The TeXbook*, p. 455 (editions after January 1990).

<sup>31</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.



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>32</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. 144 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.

```

2432 \newcommand*{\paravfootnote}[2]{%
2433   \insert\csname #1footins\endcsname
2434   \bgroup
2435     \csuse{Xnotefontsize@#1}
2436     \footsplitskips
2437     \setbox0=\vbox{\hsize=\maxdimen%
2438       \let\bidir@RTL@everypar\@empty%
2439       \noindent\csuse{Xbhooknote@#1}%
2440       \csname #1footfmt\endcsname #2{#1}}%
2441     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2442     \dp0=0pt
2443     \ht0=\csname #1footfudgefactor\endcsname\wd0
2444   %

```

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

```

2445   \if@RTL\noindent \leavevmode\fi\box0%
2446   \penalty0
2447   \egroup}
2448 %
2449 %

```

The final penalty of 0 was added here at Wayne's suggestion to avoid a weird page-breaking problem, which occurs on those occasions when  $\TeX$  attempts to split foot paragraphs. After trying out such a split (see *The TeXbook*, p. 124),  $\TeX$  inserts a penalty of  $-10000$  here, which nearly always forces the break at the end of the whole footnote paragraph (since individual notes can't be split) even when this leads to an overfull vbox. The change above results in a penalty of 0 instead which allows, but does not force, such breaks. This penalty of 0 is later removed, after page breaks have been decided, by the `\unpenalty` macro in `\makehboxofhboxes`. So it does not affect how the footnote paragraphs are typeset (the notes still have a penalty of  $-10$  between them, which is added by `\parafootfmt`).

---

<sup>32</sup>'Line Breaking', p. 610.



`\mpparavfootnote` This version is for minipages.

```

2450 \newcommand*\mpparavfootnote}[2]{%
2451   \global\setbox\@nameuse{mp#1footins}\vbox{%
2452     \unvbox\@nameuse{mp#1footins}%
2453     \csuse{Xnotefontsize@#1}
2454     \footsplitskips
2455     \setbox0=\vbox{\hsize=\maxdimen%
2456       \let\bidir@RTL@everypar\@empty%
2457       \noindent\color@begingroup%
2458       \csuse{Xbhooknote@#1}%
2459       \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2460     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2461     \dp0=\z@
2462     \ht0=\csname #1footfudgefactor\endcsname\wd0
2463     \box0
2464     \penalty0
2465   }}
2466
2467 %

```

`\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`.

```

2468 \newcommand*\Xunvxh}[2]{%
2469   \setbox0=\vbox{\unvbox#1%
2470     \global\setbox1=\lastbox}%
2471   \unhbox1
2472   \unskip           % remove \rightskip,
2473   \unskip           % remove \parfillskip,
2474   \unpenalty        % remove \penalty of 10000,
2475   \hskip\csuse{Xafternote@#2}} % but add the glue to go between the notes
2476
2477 %

```

`\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).

```

2478 \newcommand*\parafootfmt}[4]{%
2479   \Xinsertparafootsep{#4}%
2480   \ledsetnormalparstuff@common%
2481   \printlinefootnote{#1}{#4}%
2482   {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%

```



```

2483 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2484 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2485 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2486 }}%
2487 #3\penalty-10 }
2488 %

```

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.

```

2489 \newcommand*{\parafootgroup}[1]{%
2490 \unvbox\csname #1footins\endcsname
2491 \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2492 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2493 \makehboxofhboxes
2494 \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2495 \csuse{Xnotefontsize@#1}
2496 \unhbox0\par%
2497 \global\hsize=\old@hsize%
2498 }%
2499
2500 %

```

**\mpparafootgroup** The minipage version.

```

2501 \newcommand*{\mpparafootgroup}[1]{%
2502 \setXnoteswidthliketwocolumns@{#1}%
2503 \vskip\skip\@nameuse{mp#1footins}
2504 \ifl@dpairing\ifparledgroup%
2505 \leavevmode\marks\parledgroup@{begin}%
2506 \marks\parledgroup@series{#1}%
2507 \marks\parledgroup@type{Xfootnote}%
2508 \fi\fi\normalcolor
2509 \ifparledgroup%
2510 \ifl@dpairing%
2511 \else%
2512 \setXnoteswidthliketwocolumns@{#1}%
2513 \setXnotespositionliketwocolumns@{#1}%
2514 \print@Xfootnoterule{#1}%

```



```

2515 \fi%
2516 \else%
2517 \setXnoteswidthliketwocolumns@{#1}%
2518 \setXnotespositionliketwocolumns@{#1}%
2519 \print@Xfootnoterule{#1}%
2520 \fi%
2521 \unvbox\csname mp#1footins\endcsname
2522 \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2523 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2524 \makehboxofhboxes
2525 \setbox0=\hbox{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2526 \csuse{Xnotefontsize@#1}
2527 \unhbox0\par}}
2528
2529 %

```

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

```

\makehboxofhboxes 30 \newcommand*\makehboxofhboxes{\setbox0=\hbox{}}%
\removehboxes 31 \loop
2532 \unpenalty
2533 \setbox2=\lastbox
2534 \ifhbox2
2535 \setbox0=\hbox{\box2\unhbox0}%
2536 \repeat}
2537
2538 \newcommand*\removehboxes{\setbox0=\lastbox
2539 \ifhbox0{\removehboxes}\unhbox0 \fi}
2540
2541 %

```

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

```

\prevpage@num 42 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep 43 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
2544 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
2545 {\ifnumequal{\csuse{prevline#1}}{\line@num}%
2546 {\ifcsempy{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
2547 {\csuse{Xparafootsep@#1}}}%
2548 }%
2549 {\csuse{Xparafootsep@#1}}%
2550 }%
2551 {}%
2552 \global\csname #1prevpage@num\endcsname=\page@num%
2553 }

```



2554 %

## 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 a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\dosplits` `\splitoff` `\rigidbalance` macro is taken from *The TeXbook*, p. 397, with a slight change to the syntax of the arguments so that they do not depend on white space. Note also the extra unboxing in `\splitoff`, which allows the new `\vbox` to have its natural height as it goes into the alignment.

The  $\TeX$  `\line` macro has no relationship to the TeX `\line`. The  $\TeX$  equivalent is `\@@line`.

```
2555 \newcount\@k \newdimen\@h
2556 \newcommand*\rigidbalance}[3]{\setbox0=\box#1 \@k=#2 \@h=#3
2557 \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
2558 \valign{##\vfil\cr\dosplits}}
2559
2560 \newcommand*\dosplits{\ifnum\@k>0 \noalign{\hfil}\splitoff
2561 \global\advance\@k-1\cr\dosplits\fi}
2562
2563 \newcommand*\splitoff{\dimen0=\ht0
2564 \divide\dimen0 by\@k \advance\dimen0 by\@h
2565 \setbox2 \vsplit0 to \dimen0
2566 \unvbox2 }
2567
2568 %
```

### Three columns

```
\Xarrangement@threecol 2569 \newcommand*\Xarrangement@threecol}[1]{%
2570 \csgdef{series@display#1}{threecol}
2571 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
2572 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2573 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2574 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2575 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2576 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2577 \threecolfootsetup{#1}
2578 %
```

The additional setup for minipages.

```
2579 \ifnoledgroup@else
2580 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2581 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
```



```

2582 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2583 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2584 \mpthreecolfootsetup{#1}
2585 \fi
2586 }
2587
2588 %

```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 144 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.

```

2589 \newcommand*{\threecolfootsetup}[1]{%
2590   \count\csname #1footins\endcsname 333
2591   \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2592   \multiply\dimen\csname #1footins\endcsname \thr@@}
2593 %

```

`\mpthreecolfootsetup` The setup for minipages.

```

2594 \newcommand*{\mpthreecolfootsetup}[1]{%
2595   \count\csname mp#1footins\endcsname 333
2596   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
2597
2598 %

```

`\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).



```

2599 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnote}[2]{%
2600   \insert\csname #1footins\endcsname\bgroup%
2601   \noindent\csuse{Xhooknote@#1}%
2602   \csuse{Xnotefontsize@#1}%
2603   \footssplitsskip%
2604   \csname #1footfmt\endcsname #2{#1}\egroup}
2605 %

```

**\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.

```

2606 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmt}[4]{%
2607   \normal@pars%
2608   \hsize \csuse{Xhsizethreecol@#4}%
2609   \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2610   \tolerance=5000%
2611   \hangindent=\csuse{Xhangindent@#4}%
2612   \par%
2613   \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2614   \@tempdima=\parindent%
2615   \csuse{Xcolalign@#4}%
2616   \parindent=\@tempdima%
2617   \strut{\printlinefootnote{#1}{#4}}%
2618   {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafont#1|#2}{#2}}%
2619   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsempy{
Xlemmaseparator@#4}%
2620     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2621     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2622   }%
2623   #3\strut\par\allowbreak}
2624 %

```

**\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.

```

2625 \newcommand*{\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2626   \noindent\csuse{Xtxtbeforenotes@#1}}\par%
2627   \splittopskip=\ht\strutbox
2628   \expandafter
2629   \rigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}
2630 %

```



`\mpthreecolfootgroup` The setup for minipages.

```

2631 \newcommand*\mpthreecolfootgroup}[1]{%
2632   \vskip\skip\@nameuse{mp#1footins}
2633   \ifl@dpairing\ifparledgroup%
2634     \leavevmode\marks\parledgroup@{begin}%
2635     \marks\parledgroup@series{#1}%
2636     \marks\parledgroup@type{Xfootnote}%
2637   \fi\fi\normalcolor
2638   \ifparledgroup%
2639     \ifl@dpairing%
2640     \else%
2641       \setXnoteswidthliketwocolumns@{#1}%
2642       \setXnotespositionliketwocolumns@{#1}%
2643       \print@Xfootnoterule{#1}%
2644     \fi%
2645   \else%
2646     \setXnoteswidthliketwocolumns@{#1}%
2647     \setXnotespositionliketwocolumns@{#1}%
2648     \print@Xfootnoterule{#1}%
2649   \fi%
2650   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2651   \splittopskip=\ht\strutbox
2652   \expandafter
2653   \rigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2654
2655 %

```

## Two columns

```

\Xarrangement@twocol 2656 \newcommand*\Xarrangement@twocol}[1]{%
2657   \csgdef{series@display#1}{twocol}
2658   \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2659   \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
2660   \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2661   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2662   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2663   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2664   \twocolfootsetup{#1}
2665 %

```

The additional setup for minipages.

```

2666   \ifnoledgroup@else
2667     \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2668     \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2669     \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2670     \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2671     \mptwocolfootsetup{#1}
2672   \fi

```



```

2673 }
2674
2675 %

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

2676 \newcommand*\twocolfootsetup}[1]{%
2677 \count\csname #1footins\endcsname 500
2678 \csxdef{default@#1footins}{500}%Use this to confine the notes to one
side only
2679 \multiply\dimen\csname #1footins\endcsname \tw@}
2680 %

2681 \notbool{parapparatus}{\newcommand*{\newcommand}{\twocolvfootnote}[2]{%
2682 \insert\csname #1footins\endcsname\bgroup%
2683 \noindent\csuse{Xbhooknote@#1}%
2684 \csuse{Xnotefontsize@#1}%
2685 \footsplitskips%
2686 \csname #1footfmt\endcsname #2{#1}\egroup}
2687 %

2688 \notbool{parapparatus}{\newcommand*{\newcommand}{\twocolfootfmt}[4]{% 4th
arg is optional, for backward compatibility
2689 \normal@pars%
2690 \hsiz \csuse{Xhsizetwocol@#4}%
2691 \nottoggle{Xparindent@#4}{\parindent=\z@}{}%
2692 \tolerance=5000%
2693 \hangindent=\csuse{Xhangindent@#4}%
2694 \par%
2695 \everypar{\hangindent=\csuse{Xhangindent@#4}}%
2696 \@tempdima=\parindent%
2697 \csuse{Xcolalign@#4}%
2698 \parindent=\@tempdima%
2699 \strut{\printlinefootnote{#1}{#4}}%
2700 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2701 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2702 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2703 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2704 }}%
2705 #3\strut\par\allowbreak}
2706 %

2707 \newcommand*\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2708 \noindent\csuse{Xtxtbeforenotes@#1}}\par%
2709 \splittopskip=\ht\strutbox
2710 \expandafter

```



```

2711 \rigidbalance\csname #1footins\endcsname \tw@ \splittopskip}
2712
2713 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

2714 \newcommand*\mptwocolfootsetup}[1]{%
2715   \count\csname mp#1footins\endcsname 500
2716   \multiply\dimen\csname mp#1footins\endcsname \tw@}
2717 %

2718 \newcommand*\mptwocolfootgroup}[1]{%
2719   \vskip\skip\@nameuse{mp#1footins}
2720   \ifl@dpairing\ifparledgroup%
2721     \leavevmode\marks\parledgroup@{begin}%
2722     \marks\parledgroup@series{#1}%
2723     \marks\parledgroup@type{Xfootnote}%
2724   \fi\fi\normalcolor
2725   \ifparledgroup%
2726     \ifl@dpairing%
2727     \else%
2728       \setXnoteswidthliketwocolumns@{#1}%
2729       \setXnotespositionliketwocolumns@{#1}%
2730       \print@Xfootnoterule{#1}%
2731     \fi%
2732   \else%
2733     \setXnoteswidthliketwocolumns@{#1}%
2734     \setXnotespositionliketwocolumns@{#1}%
2735     \print@Xfootnoterule{#1}%
2736   \fi%
2737   {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2738   \splittopskip=\ht\strutbox
2739   \expandafter
2740   \rigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2741
2742 %

```

## 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`

`\fullstop`

`\rbracket`

The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations.



To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

The `\endashchar` macro is simply an en-dash from the normal font and is immune to changes in the surrounding font. The same goes for the full stop. These two are used in `\printlines`. The right bracket macro is the same again; it crops up in `\normalfootfmt` and the other footnote macros for controlling the format of the footnotes.

With `polyglossia`, each critical note has a `\footnote@lang` which shows the language of the lemma, and which can be used to switch the bracket from right to left.

```

2743 \def\endashchar{\textnormal{--}}
2744 \newcommand*\fullstop{\textnormal{.}}
2745 \newcommand*\rbracket{\textnormal{%
2746   \csuse{text\csuse{footnote@lang}}{%
2747     \ifluatex%
2748       \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]{\thinspace
2749 }}%
2749       \else%
2750       \thinspace}%
2751       \fi}%
2752   }}%
2753 }
2754 %
2755 %

```

### XII.7.2 Pstart number in footnote

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

```

2756 \newcommand{\printpstart}[0]{%
2757   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
2758 l@dprintingcolumns}}{%
2759     \ifledRcol%
2760       \thepstartR%
2761     \else%
2762       \thepstartL%
2763     \fi%
2764   }%
2765   \thepstart%
2766 }%
2767 %

```

### 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`.



```

2768 \newcommand{\printlinefootnote}[2]{%
2769   \def\extractline@##1|##2|##3|##4|##5|##6|##7|{##2}%
2770   \def\extractsubline@##1|##2|##3|##4|##5|##6|##7|{##3}%
2771   \def\extractendline@##1|##2|##3|##4|##5|##6|##7|{##5}%
2772   \def\extractendsubline@##1|##2|##3|##4|##5|##6|##7|{##6}%
2773   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
2774     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1| - \
extractendline@ #1| - \extractendsubline@ #1|}%
2775   }%
2776   {%
2777     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1|}%
2778   }%
2779   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2780     \hspace{\csuse{Xinplaceofnumber@#2}}%
2781   }%
2782   {%
2783     {%
2784       \iftoggle{Xnonumber@#2}{%Try if the line number must printed (by
default, yes)
2785       {%
2786         \hspace{\csuse{Xinplaceofnumber@#2}}%
2787       }%
2788       {%
2789         {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2790         {%
2791           \ifcsdef{prevline#2}%
2792             {%Be sure the \prevline exists.
2793             \ifcsequal{prevline#2}{\lineinfo@}{%Try it
2794             {%
2795               \ifcseempty{Xsymlinenum@#2}%
2796               {%
2797                 \hspace{\csuse{Xinplaceofnumber@#2}}%
2798               }%
2799               {\printsymlinefootnotearea{#2}}%
2800             }%
2801             {%
2802               \printlinefootnotearea{#1}{#2}%
2803             }%
2804           }%
2805           {%
2806             \printlinefootnotearea{#1}{#2}%
2807           }%
2808         }%
2809         {%
2810           \printlinefootnotearea{#1}{#2}%
2811         }%
2812         \csxdef{prevline#2}{\lineinfo@}%
2813       }%

```



```

2814     }%
2815     }%
2816     }%
2817 }
2818 %

```

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

```

2819 \newcommand{\printsymlinefootnotearea}[1]{%
2820   \hspace{\csuse{Xbeforexymlineenum@#1}}%
2821   \csuse{Xnotenumfont@#1}%
2822   \ifdimequal{\csuse{Xboxxymlineenum@#1}}{\z@}%
2823     {\csuse{Xsymlineenum@#1}}%
2824     {\hbox to \csuse{Xboxxymlineenum@#1}%
2825       {\csuse{Xsymlineenum@#1}\hfill}}%
2826   }%
2827   \hspace{\csuse{Xaftersymlineenum@#1}}%
2828 }%
2829 %

```

**\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.)

```

2830 \newcommand{\printlinefootnotearea}[2]{%
2831   \printXbeforenumber{#2}%
2832   \csuse{Xnotenumfont@#2}%
2833   \boxfootnotenumbers{#1}{#2}%
2834   \printXafternumber{#2}%
2835 }%
2836 %

```

**\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.

```

2837 \newcommand{\boxfootnotenumbers}[2]{%
2838   \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
2839     \printlinefootnotenumbers{#1}{#2}%
2840   }%
2841   {%
2842     \hbox to \csuse{Xboxlinenum@#2}%
2843     {%
2844       \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2845       \printlinefootnotenumbers{#1}{#2}%
2846       \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2847     }%
2848   }%

```



```
2849 }%
2850 %
```

**\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.

```
2851 \newcommand{\printlinefootnotenumbers}[2]{%
2852   \xdef\@currentseries{#2}%
2853   \ifboolexpr{%
2854     (togl{Xpstart@#2} and bool{numberpstart})%
2855     or togl{Xpstarteverytime@#2}}%
2856   {\printpstart}{}%
2857   \iftoggle{Xstanza@#2}{%
2858     \ifnumberstanza%
2859     \printstanza%
2860     \csuse{Xstanzaseparator@#2}%
2861   }%
2862   \iftoggle{Xonlypstart@#2}{\printlines#1|}%
2863 }%
2864 %
2865 %
```

**\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.)

```
2866 \newcommand{\printXbeforenumber}[1]{%
2867   \hspace{\csuse{Xbeforenumber@#1}}%
2868 }%
2869 %
```

**\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

```
2870 \newcommand{\printXafternumber}[1]{%
2871   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%
2872   \hspace{\csuse{Xafternumber@#1}}%
2873 }%
2874 %
```

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. 82: 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@pnum75 \newif\ifl@d@pnum
\ifl@d@ssub76 \newif\ifl@d@ssub
\ifl@d@elin77 \newif\ifl@d@elin
\ifl@d@esl78 \newif\ifl@d@esl
\ifl@d@dash79 \newif\ifl@d@dash
\ifl@d@Xtwolines2880 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines2881 \newif\ifl@d@Xmorethantwolines%
2882 %

\l@dparsefootsspec \l@dparsefootsspec{<spec>}{<lemma>}{<text>} parses a footnote specification. <lemma>
\l@dp@rsefootsspec and <text> are the lemma and text respectively. <spec> is the line and page num-
\l@dparsestartpage ber and lemma font specifier in \l@d@nums style format. The real work is done by
\l@dparsestartline \l@dp@rsefootsspec which defines macros holding the numeric values. Just a reminder
\l@dparsestartsub of the arguments:
\l@dparseendpage \printlines #1 | #2 | #3 | #4 | #5 | #6 | #7
\l@dparseendline \printlines start-page | line | subline | end-page | line | subline | font
\l@dparseendsub \l@dparseendsub
2883 \newcommand*{\l@dparsefootsspec}[3]{\l@dp@rsefootsspec#1|}
2884 \def\l@dp@rsefootsspec#1|#2|#3|#4|#5|#6|#7|{%
2885 \gdef\l@dparsestartpage{#1}%
2886 \gdef\l@dparsestartline{#2}%
2887 \gdef\l@dparsestartsub{#3}%
2888 \gdef\l@dparseendpage{#4}%
2889 \gdef\l@dparseendline{#5}%
2890 \gdef\l@dparseendsub{#6}%
2891 }
2892 %

```

Initialise the several number value macros.

```

2893 \def\l@dparsestartpage{0}%
2894 \def\l@dparsestartline{0}%
2895 \def\l@dparsestartsub{0}%
2896 \def\l@dparseendpage{0}%
2897 \def\l@dparseendline{0}%
2898 \def\l@dparseendsub{0}%

```



```
2899
2900 %
```

**\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`.

```
2901 \newcommand*{\setprintlines}[6]{%
2902   \l@dpnumfalse \l@ddashfalse
2903 %
```

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

```
2904   \ifbypage@
2905     \ifnum#4=#1 \else
2906       \l@dpnumtrue
2907       \l@ddashtrue
2908     \fi
2909   \fi
2910 %
```

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.

```
2911   \ifl@dpnum \l@delintrue \else \l@delinfalse \fi
2912   \ifnum#2=#5 \else
2913     \l@delintrue
2914     \l@ddashtrue
2915   \fi
2916 %
```

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

```
2917   \l@ssubfalse
2918   \ifnum#3=0 \else
2919     \l@ssubtrue
2920   \fi
2921 %
```

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.

```
2922   \l@eslfalse
2923   \ifnum#6=0 \else
2924     \ifnum#6=#3
2925       \ifl@delin \l@esltrue \else \l@eslfalse \fi
2926     \else
2927       \l@esltrue
2928       \l@ddashtrue
2929     \fi
2930   \fi%
2931 %
```



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

```

2932 \ifl@dash%
2933 \ifbool{expr{togl{fulllines@} or test{\ifcempty{Xtwolines@
@currentseries}}}}%
2934 {}%
2935 {%
2936 \setistwofollowinglines{#1}{#2}{#4}{#5}%
2937 \ifbool{expr{
2938 (%
2939 togl {Xtwolinesbutnotmore@ \@currentseries}%
2940 and not%
2941 (%
2942 bool {istwofollowinglines@}%
2943 )%
2944 )%
2945 or%
2946 (%
2947 (not test{\ifnumequal{#1}{#4}})%
2948 and togl{Xtwolinesonlyinsamepage@ \@currentseries}%
2949 )%
2950 }%
2951 {}%
2952 {%
2953 \l@d@dashfalse%
2954 \l@d@Xtwolinestrue%
2955 \l@d@elinfalse%
2956 \l@d@eslfalse%
2957 \ifcempty{Xmorethantwolines@ \@currentseries}%
2958 {}%
2959 {\ifistwofollowinglines@ \else%
2960 \l@d@Xmorethantwolinestrue%
2961 \fi%
2962 }%
2963 }%
2964 }%
2965 \fi%
2966 %

```

End of `\setprintlines`.

```

2967 }%
2968 %

```

**`\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.

```

2969 \newif\ifistwofollowinglines@%
2970 \newcommand{\setistwofollowinglines}[4]{%
2971     \ifcsdef{lastlinenumberon@#1}%
2972         {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
2973         {\numdef{\tmp}{0}}%
2974     \istwofollowinglines@false%
2975     \ifnumequal{#4-#2}{1}%
2976     {\istwofollowinglines@true}%
2977     {\ifbypage@%
2978         \ifnumequal{#3-#1}{1}%
2979         {%
2980             \ifnumequal{#2}{\tmp}%
2981             {\ifnumequal{#4}{1}{\istwofollowinglines@true}{}}%
2982             {}%
2983         }%
2984         {}%
2985     \fi%
2986 }%
2987 }%
2988 %

```

`\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`.

```

2989 \def\printlines#1|#2|#3|#4|#5|#6|#7|{%
2990     \begingroup%
2991 %

```

If we use `LuaTeX`, ensure we use good text's direction.

```

2992 \ifluatex%
2993     \edef\@tmp{\the\textdir}%
2994     \ifdefstring{\@tmp}{TLT}{\textdir TLT}%Test in order to prevent
2995     spurious space (bug #397)
2996     \fi%
2997 %

```



Decide which part of line number components we will print.

```
2997 \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
2998 %
```

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.

```
2999 \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
3000 {\bgroup}%
3001 {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
hfill}%
3002 \ifl@d@pnum #1\fullstop\fi
3003 \linenumrep{#2}
3004 \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
3005 \egroup%
3006 %
```

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

```
3007 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3008 {\bgroup}%
3009 {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
3010 \ifl@d@Xtwolines%
3011 \ifl@d@Xmorethantwolines%
3012 \csuse{Xmorethantwolines@\@currentseries}%
3013 \else%
3014 \csuse{Xtwolines@\@currentseries}%
3015 \fi%
3016 \else%
3017 \ifl@d@dash%
3018 \ifdefined\linerrangesep%
3019 \linerrangesep%
3020 \else%
3021 \csuse{Xlinerrangeseparator@\@currentseries}%
3022 \fi%
3023 \fi%
3024 \ifl@d@pnum #4\fullstop\fi%
3025 \ifl@d@elin \linenumrep{#5}\fi%
3026 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3027 \fi%
3028 \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
3029 {}%
3030 {\hfill}%Prevent underfull hbox
3031 \egroup%
3032 \endgroup%
3033 }%
3034 %
```



## XIII Familiar footnotes

### XIII.1 Adjacent footnotes

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

`\multfootsep`

```
3035 \providecommand*\multiplefootnotemarker}{3sp}
3036 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
3037
3038 %
```

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

```
3039 \providecommand*\m@mmf@prepare}{%
3040   \kern-\multiplefootnotemarker
3041   \kern\multiplefootnotemarker\relax
3042 %}
```

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

```
3043 \providecommand*\m@mmf@check}{%
3044   \ifdim\lastkern=\multiplefootnotemarker\relax
3045     \edef\@x@sf{\the\spacefactor}%
3046     \unkern
3047     \multfootsep
3048     \spacefactor\@x@sf\relax
3049   \fi}
3050
3051 %
```

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

```
3052 \@ifclassloaded{memoir}{}{%
3053 %}
```



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

```
3054 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3055 %
```

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

```
3056
3057 \patchcmd{\@footnotemark}
3058   {\nobreak}
3059   {\m@mmf@check
3060    \nobreak
3061   }
3062   {}{}
3063 \patchcmd{\@footnotemark}
3064   {\@makefnmark}
3065   {\@makefnmark
3066    \m@mmf@prepare
3067   }
3068   {}{}
3069 %
```

Finished the modifications for the non-memoir case.

```
3070 }
3071
3072 %
```

## 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.

```
3073 \pretocmd{\@footnotetext}{%
3074   \ifnumberedpar@
3075   \edtext{}{\l@dbfnote{#1}}}%
3076   \else
3077   }{}{}
3078 \apptocmd{\@footnotetext}{\fi}{}{}%
3079 %
```

`\l@dbfnote` `\l@dbfnote` adds the footnote to the insert list, and `\vl@dbfnote` calls the original `\@footnotetext`.

```
3080
3081 \newcommand{\l@dbfnote}[1]{%
3082   \ifnumberedpar@
3083   \gdef\@tag{#1\relax}%
```



```

3084 \ifledRcol%
3085 \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\
@thefnmark}}}%
3086 \to\inserts@listR
3087 \global\advance\insert@countR \@ne%
3088 \else%
3089 \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\
@thefnmark}}}%
3090 \to\inserts@list
3091 \global\advance\insert@count \@ne%
3092 \fi
3093 \fi\ignorespaces}
3094
3095 \newcommand{\vl@dbfnote}[2]{%
3096 \def\@thefnmark{#2}%
3097 \@footnotetext{#1}%
3098 }%
3099 %

```

### 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`

```

3100 \newcommand*\prebodyfootmark{%
3101 \leavevmode
3102 \ifhmode
3103 \edef\@x@sf{\the\spacefactor}%
3104 \m@mmf@check
3105 \nobreak
3106 \fi}
3107 \newcommand*\postbodyfootmark{%
3108 \m@mmf@prepare
3109 \ifhmode\spacefactor\@x@sf\fi\relax}
3110
3111 %

```

### 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.

```

3112 \newcommandx{\arrangementX}[2][1,usedefault]{%
3113 \def\do##1{%
3114 \csname arrangementX@#2\endcsname{##1}%

```



```

3115 }%
3116 \ifstrempy{#1}%
3117 {%
3118   \dolistloop{\@series}%
3119 }%
3120 {
3121   \docsvlist{#1}%
3122 }%
3123 }%
3124 %

```

### 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.

```

3125 \newcommand*{\normal@footnotemarkX}[1]{%
3126   \prebodyfootmark
3127   \@nameuse{bodyfootmark#1}%
3128   \postbodyfootmark}
3129
3130 %

```

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

```

3131 \newcommand*{\normalbodyfootmarkX}[1]{%
3132   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3133 %

```

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

```

3134 \notbool{parapparatus@}{\newcommand*{\newcommand}{\normalvfootnoteX}[2]{%
3135   \insert\@nameuse{footins#1}\bgroup
3136   \noindent\csuse{bhooknoteX@#1}%
3137   \csuse{notefontsizeX@#1}%
3138   \footsplitskips
3139   \ifl@dpairing\ifl@dpaging\else%
3140     \setnoteswidthliketwocolumnsX@{#1}%
3141   \fi\fi%
3142   \setnotesXpositionliketwocolumns@{#1}%
3143   \spaceskip=\z@skip \xspaceskip=\z@skip
3144   \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3145
3146 %

```

`\mpnormalvfootnoteX` The minipage version.



```

3147 \newcommand*{\mpnormalvfootnoteX}[2]{%
3148   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3149     \unvbox\@nameuse{mpfootins#1}
3150     \noindent\csuse{bhooknoteX@#1}%
3151     \csuse{notefontsizeX@#1}%
3152     \hsize\columnwidth
3153     \@parboxrestore
3154     \color@begingroup
3155     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3156
3157 %

```

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

```

3158 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
3159   \ifluatex%
3160     \textdir\footnote@luatextextdir%
3161     \pardir\footnote@luatexpardir%
3162     \par%
3163   \fi%
3164   \protected@edef\@currentlabel{%
3165     \@nameuse{@thefnmark#1}%
3166   }%
3167   \ledsetnormalparstuffX{#1}%
3168   \hangindent=\csuse{hangindentX@#1}%
3169   \everypar{\hangindent=\csuse{hangindentX@#1}}%
3170   {\csuse{notenunfontX@#1}\@nameuse{footfootmark#1}}\strut%
3171   #2\strut\par}}
3172
3173 %

```

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

```

3174 \newcommand*{\normalfootfootmarkX}[1]{%
3175   \textsuperscript{\@nameuse{@thefnmark#1}}}
3176
3177 %

```

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

```

3178 \newcommand*{\normalfootstartX}[1]{%
3179   \ifdimequal{Opt}{\prenotesX@}{}%
3180   {%
3181     \iftoggle{prenotesX@}{%
3182       \togglefalse{prenotesX@}%
3183       \skip\csize footins#1\endcsize=%
3184       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3185     }%

```



```

3186     {}%
3187   }%
3188   \vskip\skip\csname footins#1\endcsname%
3189   \leftskip=\z@
3190   \rightskip=\z@
3191   \ifl@dpairing\else%
3192     \hspace=\old@hspace%
3193   \fi%
3194   \setnoteswidthliketwocolumnsX@{#1}%
3195   \setnotesXpositionliketwocolumns@{#1}%
3196   \print@footnoteXrule{#1}%
3197 }%
3198
3199 %

```

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

```

3200 \let\normalfootnoteruleX=\footnoterule
3201
3202 %

```

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

```

3203 \newcommand*{\normalfootgroupX}[1]{%
3204   \unvbox\@nameuse{footins#1}%
3205   \hspace=\old@hspace%
3206 }%
3207
3208 %

```

**\mpnormalfootgroupX** The minipage version.

```

3209 \newcommand*{\mpnormalfootgroupX}[1]{%
3210   \vskip\skip\@nameuse{mpfootins#1}
3211   \ifl@dpairing\ifparledgroup%
3212     \leavevmode\marks\parledgroup@{begin}%
3213     \marks\parledgroup@series{#1}%
3214     \marks\parledgroup@type{footnoteX}%
3215   \fi\fi\normalcolor
3216   \ifparledgroup%
3217     \ifl@dpairing%
3218     \else%
3219       \setnoteswidthliketwocolumnsX@{#1}%
3220       \setnotesXpositionliketwocolumns@{#1}%
3221       \print@footnoteXrule{#1}%
3222     \fi%
3223   \else%
3224     \setnoteswidthliketwocolumnsX@{#1}%
3225     \setnotesXpositionliketwocolumns@{#1}%

```



```

3226 \print@footnoteXrule{#1}%
3227 \fi%
3228 \unvbox\@nameuse{mpfootins#1}}
3229
3230 %

```

```

\normalbfnoteX31
3232 \newcommand{\normalbfnoteX}[2]{%
3233 \ifnumberedpar@
3234 \ifledRcol%
3235 \ifluatex
3236 \footnotelang@lua[R]%
3237 \fi
3238 \@ifundefined{xpg@main@language}%if polyglossia
3239 {}%
3240 {\footnotelang@poly[R]}%
3241 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3242 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3243 \to\inserts@listR
3244 \global\advance\insert@countR \@ne%
3245 \else%
3246 \ifluatex
3247 \footnotelang@lua%
3248 \fi
3249 \@ifundefined{xpg@main@language}%if polyglossia
3250 {}%
3251 {\footnotelang@poly}%
3252 \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3253 \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3254 \to\inserts@list
3255 \global\advance\insert@count \@ne%
3256 \fi
3257 \fi\ignorespaces}
3258
3259 %

```

```

\vbfnoteX60 \newcommand{\vbfnoteX}[3]{%
3261 \@namedef{@thefnmark#1}{#3}%
3262 \@nameuse{regvfootnote#1}{#1}{#2}}
3263
3264 %

```

```

\vnumfootnoteX65 \newcommand{\vnumfootnoteX}[2]{%
3266 \ifnumberedpar@
3267 \edtext{}{\normalbfnoteX{#1}{#2}}%

```



```

3268 \else
3269   \@nameuse{regvfootnote#1}{#1}{#2}%
3270 \fi}
3271
3272 %

```

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

```

3273 \newcommand*{\arrangementX@normal}[1]{%
3274   \csgdef{series@displayX#1}{normal}
3275   \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3276   \expandafter\newcount\csname prevpage#1@num\endcsname
3277   \@namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
3278   \@namedef{bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3279   \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3280   \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
3281   \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3282   \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3283   \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3284   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3285   \count\csname footins#1\endcsname=1000
3286   \csxdef{default@footins#1}{1000}%Use to have note only for one side
3287   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3288   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3289   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3290 %

```

Additions for minipages.

```

3291 \ifnoledgroup@else%
3292   \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3293   \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3294   \count\csname mpfootins#1\endcsname=1000
3295   \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3296   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3297   \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3298 \fi
3299 }
3300
3301 %

```

### 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%2 \newcommand*{\arrangementX@twocol}[1]{%
3303   \csgdef{series@displayX#1}{twocol}
3304   \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX

```



```

3305 \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3306 \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3307 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3308 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3309 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3310 \twocolfootsetupX{#1}
3311 \ifnoledgroup@else%
3312   \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3313   \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
3314   \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3315   \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3316   \mptwocolfootsetupX{#1}
3317 \fi%
3318 }
3319
3320 %

```

`\twocolfootsetupX` `\twocolfootsetupX{<series>}`

`\mptwocolfootsetupX`

```

3321 \newcommand*{\twocolfootsetupX}[1]{%
3322   \count\csname footins#1\endcsname 500
3323   \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3324   \multiply\dimen\csname footins#1\endcsname by \tw@}
3325 \newcommand*{\mptwocolfootsetupX}[1]{%
3326   \count\csname mpfootins#1\endcsname 500
3327   \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
3328
3329 %

```

`\twocolvfootnoteX` `\twocolvfootnoteX{<series>}`

```

3330 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
3331   \insert\csname footins#1\endcsname\bgroup%
3332   \noindent\csuse{bhooknoteX@#1}%
3333   \csuse{notefontsizeX@#1}%
3334   \footsplitskips%
3335   \spaceskip=\z@skip \xspaceskip=\z@skip%
3336   \@nameuse{footfmt#1}{#1}{#2}\egroup}
3337
3338 %

```

`\twocolfootfmtX` `\twocolfootfmtX{<series>}`

```

3339 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3340   \protected@edef\@currentlabel{%
3341     \@nameuse{@thefnmark#1}%
3342   }%
3343   \normal@pars%
3344   \hangindent=\csuse{hangindentX@#1}%

```



```

3345 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3346 \hsize \csuse{hsizetwocolX@#1}%
3347 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3348 \tolerance=5000\relax%
3349 \par%
3350 \@tempdima=\parindent%
3351 \csuse{colalignX@#1}%
3352 \parindent=\@tempdima%
3353 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3354 #2\strut\par}\allowbreak}
3355
3356 %

```

```

\twocolfootgroupX \twocolfootgroupX{<series>}
\mptwocolfootgroupX
3357 \newcommand*{\twocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3358 \splittopskip=\ht\strutbox
3359 \expandafter
3360 \rigidbalance\csname footins#1\endcsname \tw@ \splittopskip}}
3361 \newcommand*{\mptwocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3362 \vskip\skip\@nameuse{mpfootins#1}
3363 \ifl@dpairing\ifparledgroup%
3364 \leavevmode\marks\parledgroup@{begin}%
3365 \marks\parledgroup@series{#1}%
3366 \marks\parledgroup@type{footnoteX}%
3367 \fi\fi\normalcolor
3368 \ifparledgroup%
3369 \ifl@dpairing%
3370 \else%
3371 \setnoteswidthliketwocolumnsX@{#1}%
3372 \setnotesXpositionliketwocolumns@{#1}%
3373 \print@footnoteXrule{#1}%
3374 \fi%
3375 \else%
3376 \setnoteswidthliketwocolumnsX@{#1}%
3377 \setnotesXpositionliketwocolumns@{#1}%
3378 \print@footnoteXrule{#1}%
3379 \fi%
3380 \splittopskip=\ht\strutbox
3381 \expandafter
3382 \rigidbalance\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3383
3384 %

```

#### 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 3385 \newcommand*{\arrangementX@threecol}[1]{%
3386 \csgdef{series@displayX#1}{threecol}
3387 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3388 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3389 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3390 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3391 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3392 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3393 \threecolfootsetupX{#1}
3394 \ifnoledgroup@else%
3395 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3396 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3397 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3398 \advance\skip\csname mpfootins#1\endcsname by \csuse{afterruleX@#1}
3399 \mpthreecolfootsetupX{#1}
3400 \fi%
3401 }
3402
3403 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX
3404 \newcommand*{\threecolfootsetupX}[1]{%
3405 \count\csname footins#1\endcsname 333
3406 \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3407 \multiply\dimen\csname footins#1\endcsname by \thr@@
3408 \newcommand*{\mpthreecolfootsetupX}[1]{%
3409 \count\csname mpfootins#1\endcsname 333
3410 \multiply\dimen\csname mpfootins#1\endcsname by \thr@@
3411
3412 %

```

```

\threecolvfootnoteX \threecolvfootnoteX{<series>}{<text>}
3413 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{
%
3414 \insert\csname footins#1\endcsname\bgroup%
3415 \noindent\csuse{bhooknoteX@#1}%
3416 \csuse{notefontsizeX@#1}
3417 \footssplitsskip%
3418 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3419
3420 %

```

```

\threecolfootfmtX \threecolfootfmtX{<series>}
3421 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3422 \protected@edef\@currentlabel{%
3423 \@nameuse{@thefnmark#1}%

```



```

3424 }%
3425 \hangindent=\csuse{hangindentX@#1}%
3426 \everypar{\hangindent=\csuse{hangindentX@#1}}%
3427 \normal@pars%
3428 \hsize \csuse{hsizethreecolX@#1}%
3429 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3430 \tolerance=5000\relax%
3431 \par%
3432 \@tempdima=\parindent%
3433 \csuse{colalignX@#1}%
3434 \parindent=\@tempdima%
3435 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3436 #2\strut\par}\allowbreak}
3437
3438 %

\threecolfootgroupX \threecolfootgroupX{<series>}
\mpthreecolfootgroupX
3439 \newcommand*{\threecolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3440 \splittopskip=\ht\strutbox
3441 \expandafter
3442 \rigidbalance\csname footins#1\endcsname \thr@@ \splittopskip}}
3443 \newcommand*{\mpthreecolfootgroupX}[1]{%
3444 \vskip\skip\@nameuse{mpfootins#1}
3445 \ifl@dpairing\ifparledgroup
3446 \leavevmode\marks\parledgroup@{begin}%
3447 \marks\parledgroup@series{#1}%
3448 \marks\parledgroup@type{footnoteX}%
3449 \fi\fi\normalcolor
3450 \ifparledgroup%
3451 \ifl@dpairing%
3452 \else%
3453 \setnoteswidthliketwocolumnsX@{#1}%
3454 \setnotesXpositionliketwocolumns@{#1}%
3455 \print@footnoteXrule{#1}%
3456 \fi%
3457 \else%
3458 \setnoteswidthliketwocolumnsX@{#1}%
3459 \setnotesXpositionliketwocolumns@{#1}%
3460 \print@footnoteXrule{#1}%
3461 \fi%
3462 \splittopskip=\ht\strutbox
3463 \expandafter
3464 \rigidbalance\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3465
3466 %

```

### XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.



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

```

3467 \newcommand*{\arrangementX@paragraph}[1]{%
3468   \csgdef{series@displayX#1}{paragraph}%
3469   \expandafter\newcount\csname #1prevpage@num\endcsname
3470   \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3471   \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
3472   \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3473   \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3474   \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3475   \count\csname footins#1\endcsname=1000
3476   \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3477   \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3478   \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3479   \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3480   \para@footsetupX{#1}
3481   \ifnoledgroup@else
3482     \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
3483     \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
3484     \count\csname mpfootins#1\endcsname=1000
3485     \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3486     \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3487     \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3488   \fi
3489 }
3490
3491 %

```

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

```

3492 \newcommand*{\para@footsetupX}[1]{\csuse{notefontsizeX@#1}
3493   \setnoteswidthliketwocolumnsX@{#1}%
3494   \dimen0=\baselineskip
3495   \multiply\dimen0 by 1024
3496   \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
%
3497   \expandafter
3498   \xdef\csname footfudgefactor#1\endcsname{%
3499     \expandafter\strip@pt\dimen0 }}
3500
3501 %

```

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

```

3502 \newcommand*{\parafootstartX}[1]{%
3503   \ifdimequal{0pt}{\prenotesX@}{}%
3504   {%
3505     \iftoggle{prenotesX@}{%
3506       \togglefalse{prenotesX@}%

```



```

3507         \skip\csname footins#1\endcsname=%
3508         \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3509         }%
3510     {}%
3511 }%
3512 \leftskip=\z@
3513 \rightskip=\z@
3514 \nottoggle{parindentX@#1}{\parindent=\z@}{}%
3515 \vskip\skip\@nameuse{footins#1}%
3516 \setnoteswidthliketwocolumnsX@{#1}%
3517 \setnotesXpositionliketwocolumns@{#1}%
3518 \print@footnoteXrule{#1}%
3519 }
3520
3521 %

```

```

\para@vfootnoteX \para@vfootnoteX{<series>}{<text>}
\mppara@vfootnoteX
3522 \newcommand*{\para@vfootnoteX}[2]{%
3523     \insert\csname footins#1\endcsname
3524     \bgroup
3525         \csuse{notefontsizeX@#1}
3526         \footsplitskips
3527         \setbox0=\vbox{\hsize=\maxdimen%
3528             \let\bid@RTL@everypar\@empty%
3529             \noindent\csuse{bhooknoteX@#1}%
3530             \@nameuse{footfmt#1}{#1}{#2}}%
3531         \setbox0=\hbox{\unvvhX{0}{#1}}%
3532         \dp0=\z@
3533         \ht0=\csname footfudgefactor#1\endcsname\wd0
3534         \box0
3535         \penalty0
3536     \egroup}
3537 \newcommand*{\mppara@vfootnoteX}[2]{%
3538     \global\setbox\@nameuse{mpfootins#1}\vbox{%
3539         \unvbox\@nameuse{mpfootins#1}
3540         \csuse{notefontsizeX@#1}
3541         \footsplitskips
3542         \setbox0=\vbox{\hsize=\maxdimen%
3543             \let\bid@RTL@everypar\@empty%
3544             \noindent\color@begingroup%
3545             \csuse{bhooknoteX@#1}%
3546             \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3547         \setbox0=\hbox{\unvvhX{0}{#1}}%
3548         \dp0=\z@
3549         \ht0=\csname footfudgefactor#1\endcsname\wd0
3550         \box0
3551         \penalty0}}
3552
3553 %

```



```

\unvxhX \newcommand*{\unvxhX}[2]{% 2th is optional for retro-compatibility
3555 \setbox0=\vbox{\unvbox#1%
3556 \global\setbox1=\lastbox}%
3557 \unhbox1
3558 \unskip           % remove \rightskip,
3559 \unskip           % remove \parfillskip,
3560 \unpenalty        % remove \penalty of 10000,
3561 \hskip\csuse{afternoteX@#2}} % but add the glue to go between the notes
3562
3563 %

```

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

```

3564 \newcommand*{\parafootfmtX}[2]{%
3565 \protected@edef\@currentlabel{%
3566 \@nameuse{@thefnmark#1}%
3567 }%
3568 \insertparafootsepX{#1}%
3569 \ledsetnormalparstuff@common%
3570 {\csuse{notenumfontX@#1}%
3571 \csuse{notenumfontX@#1}%
3572 \@nameuse{footfootmark#1}%
3573 \strut%
3574 #2\penalty-10}}
3575
3576 %

```

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

```

3577 \newcommand*{\para@footgroupX}[1]{%
3578 \unvbox\csname footins#1\endcsname
3579 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3580 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3581 \makehboxofhboxes
3582 \setbox0=\hbox{\unhbox0 \removehboxes}%
3583 \csuse{notefontsizeX@#1}
3584 \unhbox0\par}
3585 \newcommand*{\mppara@footgroupX}[1]{%
3586 \setnoteswidthliketwocolumnsX@{#1}%
3587 \vskip\skip\@nameuse{mpfootins#1}
3588 \ifl@dpairing\ifparledgroup
3589 \leavevmode%
3590 \leavevmode\marks\parledgroup@{begin}%
3591 \marks\parledgroup@series{#1}%
3592 \marks\parledgroup@type{footnoteX}%
3593 \fi\fi\normalcolor
3594 \ifparledgroup%
3595 \ifl@dpairing%
3596 \else%
3597 \setnoteswidthliketwocolumnsX@{#1}%

```



```

3598 \setnotesXpositionliketwocolumns@{#1}%
3599 \print@footnoteXrule{#1}%
3600 \fi%
3601 \else%
3602 \setnoteswidthliketwocolumnsX@{#1}%
3603 \setnotesXpositionliketwocolumns@{#1}%
3604 \print@footnoteXrule{#1}%
3605 \fi%
3606 \unvbox\csname mpfootins#1\endcsname
3607 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3608 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3609 \makehboxofhboxes
3610 \setbox0=\hbox{\unhbox0 \removehboxes}%
3611 \csuse{notefontsizeX@#1}
3612 \unhbox0\par}}
3613
3614 %

```

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

```

\prevpage@num15 \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep16 \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3617 {\csuse{parafootsepX@#1}}%
3618 {}}%
3619 }
3620 %

```

## 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 `LuaLaTeX`.

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.

```

dsetnormalparstuff@common21 \newcommand*{\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff22 \ifluatex%
\ledsetnormalparstuffX23 \textdir\footnote@luatextextdir%
3624 \pardir\footnote@luatexpardir%
3625 \fi%

```



```

3626 \csuse{\csuse{footnote@dir}}%
3627 \normal@pars%
3628 \parfillskip \z@ \@plus 1fil}%
3629
3630 \newcommand*{\Xledsetnormalparstuff}[1]{%
3631 \ledsetnormalparstuff@common%
3632 \nottoggle{Xparindent@#1}{\parindent=\z@}{\hspace{\parindent}}%
3633 }%
3634
3635 \newcommand*{\ledsetnormalparstuffX}[1]{%
3636 \ledsetnormalparstuff@common%
3637 \nottoggle{parindentX@#1}{\parindent=\z@}{\hspace{\parindent}}%
3638 }%
3639 %

```

## 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@`

```

3640
3641 \newdimen\old@hsize%
3642 \AtBeginDocument{\old@hsize=\hsize}%
3643
3644 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3645 \global\let\hsize@fornote=\hsize%
3646 \global\old@hsize=\hsize%
3647 \iftoggle{Xnoteswidthliketwocolumns@#1}%
3648 {%
3649 \csuse{setwidthliketwocolumns@\columns@position}%
3650 \global\let\hsize@fornote=\hsize%
3651 }%
3652 }%
3653 \let\hsize=\hsize@fornote%
3654 \let\columnwidth=\hsize@fornote%
3655 }%
3656
3657 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3658 \global\let\hsize@fornote=\hsize%
3659 \global\old@hsize=\hsize%
3660 \iftoggle{noteswidthliketwocolumnsX@#1}%
3661 {%
3662 \csuse{setwidthliketwocolumns@\columns@position}%
3663 \global\let\hsize@fornote=\hsize%

```



```

3664 }%
3665 {}%
3666 \let\hsize=\hsize@fornote%
3667 \let\columnwidth=\hsize@fornote%
3668 }%
3669 %
3670 %

```

`\setnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sens without `reledpar`.

```

3671 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3672 \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3673 \csuse{setnotespositionliketwocolumns@\columns@position}%
3674 }{}%
3675 }%
3676 %
3677 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3678 \iftoggle{noteswidthliketwocolumnsX@#1}{%
3679 \csuse{setnotespositionliketwocolumns@\columns@position}%
3680 }{}%
3681 }%
3682 %
3683 %

```

## 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
3684 \def\@fnpos{familiar-critical}
3685 \def\@mpfnpos{critical-familiar}
3686 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3687 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3688 %

```

## 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@Xfootnoterule89 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnoterule90 \vskip-\csuse{Xafterrule@#1}%Because count in \dimen\csuse{#1footins}
3691 \nointerlineskip%

```



```

3692 \moveleft-\leftskip\ vbox{\csuse{#1footnoterule}}}%
3693 \nointerlineskip%
3694 \vskip\csuse{Xafterterrule@#1}%
3695 }%
3696
3697 \newcommand{\print@footnoterule}[1]{%
3698 \vskip-\csuse{afterterruleX@#1}%Because count in \dimen\csuse{footins#1}
3699 \nointerlineskip%
3700 \moveleft-\leftskip\ vbox{\csuse{footnoterule#1}}}%
3701 \nointerlineskip%
3702 \vskip\csuse{afterterruleX@#1}%
3703 }%
3704
3705 %

```

## 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 `\Xafterterrule` does not depend on whether the series is the first one of the page or not. So we use its normal value for each series.

And now, implementation !



### XVIII.0.2 User level command

`\preXnotes@` If user redefines `\preXnotes@`, via `\preXnotes` to a value greater than 0 pt, this skip will be added before first series notes instead of the notes skip.

```

3706 \newtoggle{preXnotes@}
3707 \toggletrue{preXnotes@}
3708 \newcommand{\preXnotes@}{Opt}
3709 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3710 %

```

The same, but for familiar footnotes.

```

\preXnotes 11 \newtoggle{prenotesX@}
\preXnotes 12 \toggletrue{prenotesX@}
3713 \newcommand{\prenotesX@}{Opt}
3714 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
3715 %

```

### XVIII.0.3 Internal commands

```

firstXseries@ 16 \gdef\firstXseries@{}
prepare@preXnotes 17 \newcommand{\prepare@preXnotes}[1]{%
3718 \ifdimequal{0pt}{\preXnotes@}%
3719 {}%
3720 {%
3721 \IfStrEq{\firstXseries@}{}{%
3722 \global\skip\csuse{#1footins}=\preXnotes@%
3723 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3724 \gdef\firstXseries@{#1}%
3725 }%
3726 {%
3727 \ifseriesbefore{#1}{\firstXseries@}%
3728 {%
3729 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3730 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
3731 \gdef\firstXseries@{#1}%
3732 }%
3733 {}%
3734 }%
3735 }%
3736 }
3737 %

```

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



```

firstseriesX@38 \gdef\firstseriesX@{}
prepare@prenotesX39 \newcommand{\prepare@prenotesX}[1]{%
3740 \ifdimequal{0pt}{\prenotesX@}%
3741 {}%
3742 {%
3743 \IfStrEq{\firstseriesX@}{-}{%
3744 \global\skip\csuse{footins#1}=\prenotesX@%
3745 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3746 \gdef\firstseriesX@{#1}%
3747 }%
3748 {%
3749 \ifseriesbefore{#1}{\firstseriesX@}%
3750 {%
3751 \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
3752 \global\advance\skip\csname footins#1\endcsname by\csuse{afterruleX@
#1}%
3753 \gdef\firstXseries@{#1}%
3754 }%
3755 {}%
3756 }%
3757 }%
3758 }
3759 %

```

## XIX Endnotes

First, check the noend option.

```

3760 \ifbool{noend@}{-}{%Used instead of \ifnoend@ to prevent expansion problem
3761 %

```

**\l@dend@open** \l@dend@open and \l@dend@close are the macros that are used to open and close the endnote file. Note that all our writing to this file is \immediate: all page and line numbers for the endnotes are generated by the same mechanism we use for the footnotes, so that there is no need to defer any writing to catch information from the output routine. The argument of these two command is the series letter.

```

3762 \newcommand{\l@dend@open}[1]{%
3763 \global\booltrue{l@dend@#1}%
3764 \expandafter\immediate%
3765 \expandafter\openout%
3766 \csname l@d@#1end\endcsname%
3767 =\jobname.#1end\relax%
3768 }%
3769 \newcommand{\l@dend@close}[1]{%
3770 \global\boolfalse{l@dend@#1}%
3771 \expandafter\immediate%

```



```

3772 \expandafter\closeout\csname l@d@#1end\endcsname%
3773 }%
3774
3775 %

```

**\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.

```

3776 \newcommand{\l@dend@stuff}{%
3777 \def\do##1{%
3778 \ifbool{l@dend@##1}{}%
3779 {\l@dend@open{##1}}%
3780 \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{\
string\l@d@section{\the\section@num}}%
3781 }%
3782 \dolistloop{\@series}%
3783 }%
3784
3785 %

```

**\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.

```

3786 \global\newbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5{%
3787 \hangindent=\csuse{Xendhangindent@#4}%
3788 \ifXendinsertsep@%
3789 \hskip\csuse{Xendafternote@#4}%
3790 \csuse{Xendsep@#4}%
3791 \else%
3792 \iftoggle{Xendparagraph@#4}%
3793 {\global\Xendinsertsep@true}%
3794 }%
3795 \fi%
3796 \xdef\@currentseries{#4}%
3797 \def\do##1{%
3798 \setkeys[mac]{truefootnoteoption}{##1}%

```



```

3799 }%
3800 \notblank{#5}{\docsvlist{#5}}{}}%
3801 \csuse{Xendbhooknote@#4}%
3802 \csuse{Xendnotefontsize@#4}%
3803 \printlineendnote{#1}{#4}%
3804 \nottoggle{Xendlemmadisablefontselection@#4}%
3805   {\select@lemmafont#1|#2}%
3806   {#2}%
3807 \ifboolexpr{%
3808   togl {nosep@}%
3809   or test{\ifcsemtty{Xendlemmaseparator@#4}}}%
3810 }%
3811 {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
3812 {\nobreak%
3813   \hskip\csuse{Xendbeforelemmaseparator@#4}%
3814   \csuse{Xendlemmaseparator@#4}%
3815   \hskip\csuse{Xendafterlemmaseparator@#4}%
3816 }%
3817 #3%
3818 \nottoggle{Xendparagraph@#4}{\par}{}}%
3819 \def\do##1{%
3820   \setkeys[mac]{falsefootnoteoption}{##1}%
3821 }%
3822 \notblank{#5}{\docsvlist{#5}}{}}%
3823 }}%
3824
3825 \let\l@d@section=\@gobble
3826
3827 %

```

**\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.

```

3828 \newcommand{\printlineendnote}[2]{%
3829   \l@dp@rsefootspec#1|}%
3830   \iftoggle{Xendnumberonlyfirstintwolines@#2}{%
3831     \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
3832       \l@dparsedstartsub - \l@dparsedendpage - \l@dparsedendline - \
3833       \l@dparsedendsub}%
3834     }%
3835     {\
3836       \edef\lineinfo@{\l@dparsedstartpage - \l@dparsedstartline - \
3837       \l@dparsedstartsub}%
3838     }%
3839     \ifboolexpr{%
3840       togl {nonum@}%
3841       or togl {Xendnonumber@#2}}%
3842     {\hspace{\csuse{Xendinplaceofnumber@#2}}}%

```



```

3841 {%
3842 \iftoggle{Xendnumberonlyfirstinline@#2}%
3843 {\ifcsdef{prevendline#2}%
3844 {\ifcsequal{prevendline#2}{lineinfo@}%
3845 {%
3846 \csuse{Xendbhookinplaceofnumber@#2}%
3847 \ifcsequal{Xendsymmlinenumber@#2}%
3848 {\hspace{\csuse{Xendinplaceofnumber@#2}}}%
3849 {\printsymmlineendnotearea{#2}}%
3850 \csuse{Xendahookinplaceofnumber@#2}%
3851 }%
3852 {\printlineendnotearea{#1}{#2}}}%
3853 {\printlineendnotearea{#1}{#2}}}%
3854 }%
3855 {\printlineendnotearea{#1}{#2}}%We keep every time line
3856 \csxdef{prevendline#2}{\lineinfo@}%
3857 }%
3858 }%
3859 %

```

```

\printsymmlineendnotearea \newcommand{\printsymmlineendnotearea}[1]{%
3861 \hspace{\csuse{Xendbeforemsymmlinenumber@#1}}%
3862 \csuse{Xendnotenumfont@#1}%
3863 \ifdimequal{\csuse{Xendboxsymmlinenumber@#1}}{\z@}%
3864 {\csuse{Xendsymmlinenumber@#1}}%
3865 {\hbox to \csuse{Xendboxsymmlinenumber@#1}%
3866 {\csuse{Xendsymmlinenumber@#1}\hfill}%
3867 }%
3868 \hspace{\csuse{Xendaftersymmlinenumber@#1}}%
3869 }%
3870 %

```

**\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.)

```

3871 \newcommand{\printlineendnotearea}[2]{%
3872 \csuse{Xendbhooklinenumber@#2}%
3873 \hspace{\csuse{Xendbeforenumber@#2}}%
3874 \bgroup%
3875 \csuse{Xendnotenumfont@#2}%
3876 \ifdimequal{\csuse{Xendboxlinenumber@#2}}{\Opt}%
3877 {\printendlines#1}%
3878 {\leavevmode%
3879 \hbox to \csuse{Xendboxlinenumber@#2}%
3880 {%
3881 \IfSubStr{RC}{\csuse{Xendboxlinenumberalign@#2}}{\hfill}{}%
3882 \printendlines#1}%

```



```

3883 \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}%
3884 }}%
3885 \egroup%
3886 \hspace{\csuse{Xendafternumber@#2}}%
3887 \csuse{Xendahooklinenum@#2}%
3888 }%
3889 %

```

**\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.

```

3890 \newcommand*{\setprintendlines}[6]{%
3891 \l@dpnumfalse \l@ddashfalse
3892 \ifnum#4=#1 \else
3893 \l@dpnumtrue
3894 \l@ddashtrue
3895 \fi
3896 %

```

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.

```

3897 \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
3898 \ifnum#2=#5 \else
3899 \l@d@elintrue
3900 \l@ddashtrue
3901 \fi
3902 %

```

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

```

3903 \l@d@ssubfalse
3904 \ifnum#3=0 \else
3905 \l@d@ssubtrue
3906 \fi
3907 %

```

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.

```

3908 \l@d@eslfalse
3909 \ifnum#6=0 \else
3910 \ifnum#6=#3
3911 \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3912 \else

```



```

3913         \l@d@esltrue
3914         \l@d@dashtrue
3915     \fi
3916 \fi%
3917 %

3918 \ifl@d@dash%
3919     \ifbool{expr{togl{fulllines@} or test{\ifcsemt{Xendtwolines@
@currentseries}}}}%
3920     {}%
3921     {%
3922     \setistwofollowinglines{#1}{#2}{#4}{#5}%
3923     \ifbool{expr{%
3924         (%
3925             togl{Xendtwolinesbutnotmore@ \@currentseries}%
3926             and not%
3927             (%
3928                 bool{istwofollowinglines@}%
3929             )%
3930         )%
3931         or%
3932         (%
3933             (not test{\ifnumequal{#1}{#4}})%
3934             and togl{Xendtwolinesonlyinsamepage@ \@currentseries}%
3935         )%
3936     }%
3937     {}%
3938     {%
3939     \l@d@dashfalse%
3940     \l@d@Xtwolinestrue%
3941     \l@d@elinfalse%
3942     \l@d@eslfalse%
3943     \ifcsemt{Xendmorethantwolines@ \@currentseries}%
3944     {%
3945     \ifistwofollowinglines@ \else%
3946     \l@d@Xmorethantwolinestrue%
3947     \fi%
3948     }%
3949     }%
3950     }%
3951 \fi%
3952 %

End of \setprintendlines.

3953 }%
3954 %

```

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

```

3955 \def\printendlines#1|#2|#3|#4|#5|#6|#7|{\begingroup

```



```

3956 \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
3957 %

```

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.

```

3958 \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
3959 {\bgroup}%
3960 {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%
3961 \printnpnum{#1}%
3962 \linenumrep{#2}%
3963 \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
3964 \egroup%
3965 %

```

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

```

3966 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3967 {\bgroup}%
3968 {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
3969 \ifl@d@Xtwolines%
3970 \ifl@d@Xmorethantwolines%
3971 \csuse{Xendmorethantwolines@\@currentseries}%
3972 \else%
3973 \csuse{Xendtwolines@\@currentseries}%
3974 \fi%
3975 \else%
3976 \ifl@d@dash%
3977 \ifdefined\linerrangesep%
3978 \linerrangesep%
3979 \else%
3980 \csuse{Xendlinerrangeseparator@\@currentseries}%
3981 \fi%
3982 \fi%
3983 \ifl@d@pnum \printnpnum{#4}\fi%
3984 \ifl@d@elin \linenumrep{#5}\fi%
3985 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3986 \fi%
3987 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3988 {}%
3989 {\hfill}%Prevent underfull hbox
3990 \egroup%
3991 \endgroup%
3992 }%
3993
3994 %

```



`\printnpnum` A macro to print a page number in an endnote.

```
3995 \newcommand*{\printnpnum}[1]{p.#1) }
3996
3997 %
```

`\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.

```
3998 \newif\ifXendinsertsep@%
3999 \newcommand*{\doendnotes}[1]{%
4000   \l@dend@close{#1}%
4001   \beginngroup
4002     \makeatletter
4003     \expandafter\let\csname #1end\endcsname=\endprint
4004     \input\jobname.#1end%
4005     \global\Xendinsertsep@false%
4006   \endgroup}
4007 %
```

`\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.

```
4008 \newcommand*{\doendnotesbysection}[1]{%
4009   \l@dend@close{#1}%
4010   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
4011   \beginngroup%
4012     \makeatletter%
4013     \def\l@d@section##1{%
4014       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
4015       {\cslet{#1end}{\endprint}}%
4016       {\cslet{#1end}{\@gobblefive}}%
4017     }%
4018     \input\jobname.#1end%
4019     \global\Xendinsertsep@false%
4020   \endgroup%
4021 }%
4022 %
```

End of section for end notes

```
4023 }%
4024 %
```



## XX Generate series of notes

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

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

```
4025 \newcommand{\newseries}[1]{%
4026     \def\do##1{\newseries@{##1}}%
4027     \docsvlist{#1}
4028 }
4029 %
```

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

```
4030 \newcommand{\@series}{}
4031 %
```

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

```
\newseries@32 \newcommand{\newseries@}[1]{
4033 %
```

### XX.1 Test if series is still existing

```
4034 \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
4035 {%
4036 %
```

### XX.2 Init specific to reledpar

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

```
4037 \ifdefined\newseries@par%
4038     \newseries@par{#1}%
4039 \fi%
4040 %
```

### XX.3 For critical footnotes

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

```
4041 \unless\ifnocritical@
4042 %
```



## XX.3.1 Options

```

4043 \newtoggle{Xparindent@#1}
4044 \newtoggle{Xlemmadisablefontselection@#1}
4045 \csgdef{Xhangindent@#1}{Opt}%
4046 \csgdef{Xragged@#1}{}%
4047 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
4048 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
4049 \csgdef{Xcolalign@#1}{\raggedright}%
4050 \csgdef{Xnotenumfont@#1}{\normalfont}%
4051 \csgdef{Xnotefontsize@#1}{\footnotesize}%
4052 \csgdef{Xbhooknote@#1}{}%
4053
4054 \csgdef{Xboxlinenum@#1}{Opt}%
4055 \csgdef{Xboxlinenumalign@#1}{L}%
4056
4057 \csgdef{Xboxstartlinenum@#1}{Opt}%
4058 \csgdef{Xboxendlinenum@#1}{Opt}%
4059
4060 \csgdef{Xboxsymlinenum@#1}{Opt}%
4061 \newtoggle{Xnumberonlyfirstinline@#1}%
4062 \newtoggle{Xnumberonlyfirstintwolines@#1}%
4063 \csgdef{Xtwolines@#1}{}%
4064 \csgdef{Xmorethantwolines@#1}{}%
4065 \newtoggle{Xtwolinesbutnotmore@#1}%
4066 \newtoggle{Xtwolinesonlyinsamepage@#1}%
4067 \newtoggle{Xonlypstart@#1}%
4068 \newtoggle{Xpstarteverytime@#1}%
4069 \newtoggle{Xpstart@#1}%
4070 \newtoggle{Xstanza@#1}%
4071 \csgdef{Xstanzaseparator@#1}{}%
4072 \csgdef{Xsymlinenum@#1}{}%
4073 \newtoggle{Xnonumber@#1}%
4074 \csgdef{Xbeforenumber@#1}{Opt}%
4075 \csgdef{Xafternumber@#1}{0.5em}%
4076 \newtoggle{Xnonbreakableafternumber@#1}%
4077 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
4078 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
4079 \csgdef{Xinplaceofnumber@#1}{1em}%
4080 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
4081 \csgdef{Xbeforelemmaseparator@#1}{0em}%
4082 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
4083 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
4084 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}%
4085 \csgdef{Xafterrule@#1}{Opt}%
4086 \csgdef{Xtxtbeforenotes@#1}{}%
4087 \csgdef{Xmaxhnotes@#1}{0.8\vsizer}%
4088 \newtoggle{Xnoteswidthliketwocolumns@#1}%
4089 \csgdef{Xparafootsep@#1}{}%
4090 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}

```



```

4091 \csgdef{Xlinerangeseparator@#1}{\endashchar}%
4092 %

```

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

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

```

4093 \expandafter\newinsert\csname #1footins\endcsname%
4094 \unless\ifnoledgroup%
4095 \expandafter\newinsert\csname mp#1footins\endcsname%
4096 \fi%
4097 %

```

### 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.

```

4098 \global\newcommand{\parapparatus@}{\expandafter\newcommand\expandafter
*}{\expandafter\newcommand}\csname #1footnote\endcsname[2][]{%
4099 \ifnum\@edtext@level>0%
4100 \begingroup%
4101 \newcommand{\content}{##2}%
4102 \ifnumberedpar%
4103 \ifledRcol%
4104 \ifluatex%
4105 \footnotelang@lua[R]%
4106 \fi%
4107 \@ifundefined{xpg@main@language}%if polyglossia
4108 {}%
4109 {\footnotelang@poly[R]}%
4110 \footnoteoptions@{R}{##1}{true}%
4111 \xright@appenditem{%
4112 \noexpand\prepare@preXnotes{#1}%
4113 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4114 \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.
4115 \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4116 \noexpand\csuse{v#1footnote}{#1}%
4117 {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}%
%
4118 }\to\inserts@listR
4119 \footnoteoptions@{R}{##1}{false}%
4120 \global\advance\insert@countR \@ne%
4121 \else%
4122 \ifluatex%
4123 \footnotelang@lua%
4124 \fi%

```



```

4125         \ifundefined{xpg@main@language}%if polyglossia
4126         {%
4127         {\footnotelang@poly}%
4128         \footnoteoptions@{L}{##1}{true}%
4129         \xright@appenditem{%
4130         \noexpand\prepare@preXnotes{#1}%
4131         \noexpand\prepare@edindex@fornote{\l@d@nums}%
4132         \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.
4133         \ifl@pairing%
4134         \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4135         \fi%
4136         \noexpand\csuse{v#1footnote}{#1}%
4137         {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}
%
4138         }\to\inserts@list
4139         \global\advance\insert@count \@ne%
4140         \footnoteoptions@{L}{##1}{false}%
4141         \fi
4142         \else
4143         \csuse{v#1footnote}{#1}{\{0|0|0|0|0|0|0|0\}}{##1}%
4144         \fi%
4145         \endgroup%
4146         \else%
4147         \led@err@FootnoteWithoutEdtext%
4148         \fi%
4149         \ignorespaces%
4150         }
4151         %

```

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

```

4152         \global\csletcs{#1@@footnote}{#1footnote}
4153         %

```

### XX.3.4 Set standard display

```

4154         \Xarrangement@normal{#1}%
4155         %

```

End of for critical footnotes.

```

4156         \fi
4157         %

```

## XX.4 For familiar footnotes

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



```

4158 \unless\ifnofamiliar@
4159 %

```

### XX.4.1 Options

```

4160 \newtoggle{parindentX@#1}
4161 \csgdef{hangindentX@#1}{Opt}%
4162 \csgdef{raggedX@#1}{}%
4163 \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
4164 \csgdef{hsizethreecolX@#1}{.3 \hsize}%
4165 \csgdef{colalignX@#1}{\raggedright}%
4166 \csgdef{notenumfontX@#1}{\normalfont}%
4167 \csgdef{notefontsizeX@#1}{\footnotesize}%
4168 \csgdef{bhooknoteX@#1}{}%
4169 \csgdef{afterruleX@#1}{Opt}
4170 \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
4171 \csgdef{maxhnotesX@#1}{0.8\vsize}%
4172 \newtoggle{noteswidthliketwocolumnsX@#1}%
4173 \csgdef{parafootsepX@#1}{}%
4174 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4175 % End of for familiar footnotes.
4176 % \subsubsection{Create inserts, needed to add notes in foot}
4177 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4178 % \begin{macrocode}
4179 \expandafter\newinsert\csname footins#1\endcsname%
4180 \unless\ifnoledgroup%
4181 \expandafter\newinsert\csname mpfootins#1\endcsname%
4182 \fi%
4183 %

```

### 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.

```

4184 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4185 \begingroup%
4186 \prepare@prenotesX{#1}%
4187 \newcommand{\content}{##1}%
4188 \stepcounter{footnote#1}%
4189 \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4190 \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4191 \csuse{@footnotemark#1}}%
4192 {}%
4193 \ifluatex%
4194 \xdef\footnote@luatextextdir{\the\textdir}%
4195 \xdef\footnote@luatexpardir{\the\pardir}%
4196 \fi%
4197

```



```

4198         \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
4199         \endgroup%
4200     }
4201 %

```

Then define the counters.

```

4202     \newcounter{footnote#1}
4203     \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4204 %

```

Do not forget to initialize series

```

4205     \arrangementX@normal{#1}%
4206     \fi
4207 %

```

## XX.5 The endnotes

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

```

4208     \unless\ifnoend@
4209 %

```

### XX.5.1 The auxiliary file

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

```

4210     \expandafter\newwrite\csname l@d@#1end\endcsname%
4211     \expandafter\newif\csname ifl@dend@#1\endcsname%
4212 %

```

### 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.

```

4213
4214     \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
usedefault]{%
4215         \bgroup%
4216         \newlinechar='40%
4217         \global\@noneed@Footnotetrue%
4218         \newcommand{\content}{##2}%

```



```

4219 \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
4220 \expandafter\string\csname #1end\endcsname%
4221 {\ifnumberedpar@l@d@nums\fi}%
4222 {\ifnumberedpar@\expandonce\@tag\fi}%
4223 {\expandonce\content}%
4224 {\#1}%
4225 {\unexpanded{##1}}%
4226 \@percentchar%
4227 }%
4228 \egroup%
4229 \ignorespaces%
4230 }%
4231 %

```

\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.

```

4232
4233 \global\cslet{#1end}{\@gobblefive}
4234 %

```

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

```

4235 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4236 %

```

### XX.5.3 The options

```

4237 \csgdef{Xendtwolines@#1}{}%
4238 \csgdef{Xendmoreethantwolines@#1}{}%
4239 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4240 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4241 \newtoggle{Xendlemmadisablefontselection@#1}%
4242 \csgdef{Xendnotenumfont@#1}{\normalfont}%
4243 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4244 \csgdef{Xendbhooknote@#1}{}%
4245
4246 \csgdef{Xendbeforenumber@#1}{0pt}
4247 \csgdef{Xendafternumber@#1}{0.5em}
4248
4249 \csgdef{Xendboxlinenum@#1}{0pt}%
4250 \csgdef{Xendboxlinenumalign@#1}{L}%
4251
4252 \csgdef{Xendboxstartlinenum@#1}{0pt}%
4253 \csgdef{Xendboxendlinenum@#1}{0pt}%
4254

```



```

4255 \csgdef{Xendlemmaseparator@#1}{}%
4256 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4257 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4258 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4259
4260 \newtoggle{Xendparagraph@#1}%
4261 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4262 \csgdef{Xendsep@#1}{}%
4263
4264 \csgdef{Xendinplaceofnumber@#1}{0pt}%
4265 \newtoggle{Xendnonumber@#1}%
4266
4267 \csgdef{Xendhangindent@#1}{0pt}%
4268 \newtoggle{Xendnumberonlyfirstinline@#1}%
4269 \newtoggle{Xendnumberonlyfirstintwolines@#1}%
4270
4271 \csgdef{Xendbeforesymlinenum@#1}{\csuse{Xendbeforenumber@#1}}%
4272 \csgdef{Xendaftersymlinenum@#1}{\csuse{Xendafternumber@#1}}%
4273 \csgdef{Xendsymlinenum@#1}{}%
4274 \csgdef{Xendboxsymlinenum@#1}{0pt}%
4275
4276 \csgdef{Xendbhooklinenum@#1}{}%
4277 \csgdef{Xendehooklinenum@#1}{}%
4278 \csgdef{Xendbhookinplaceofnumber@#1}{}%
4279 \csgdef{Xendehookinplaceofnumber@#1}{}%
4280
4281 \csgdef{Xendlinangeseparator@#1}{\endashchar}%
4282 %

```

End of endnotes declaration

```

4283 \fi%
4284 %

```

Dump series in \@series

```

4285 \listxadd{\@series}{#1}
4286 }
4287 }% End of \newseries
4288 %

```

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

```

4289 \expandafter\newseries\expandafter{\default@series}
4290 %

```

## 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.



```

4291 \newcommand{\seriesatbegin}[1]{%
4292   \StrDel{\@series}{#1}[\@series]%
4293   \edef\@new{%
4294     \listadd{\@new}{#1}%
4295     \listadd{\@new}{\@series}%
4296     \xdef\@series{\@new}%
4297   }
4298   %

```

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

```

4299 \newcommand{\seriesatend}[1]{%
4300   \StrDel{\@series}{#1}[\@series]%
4301   \edef\@new{%
4302     \listadd{\@new}{\@series}%
4303     \listadd{\@new}{#1}%
4304     \xdef\@series{\@new}%
4305   }
4306   %

```

## XXI.2 Test series order

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

```

4307 \newcommand{\ifseriesbefore}[4]{%
4308   \StrPosition{\@series}{#1}[\@first]%
4309   \StrPosition{\@series}{#2}[\@second]%
4310   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4311 }
4312 %

```

### XXI.2.1 Get the first series

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

```

\@getfirstseries13 \newcommand{\@getfirstseries}{%
4314   \ifdefempty{\@series}%
4315   {\xdef\@firstseries{}}%
4316   {\StrChar{\@series}{1}[\@firstseries]}%
4317 }%
4318 %

```

## 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`.

```

4319 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4320   \def\do##1{%
4321     \global\settoggle{#2@##1}{#3}%
4322     \ifstrequal{#4}{critical}{
4323       \csuse{Xarrangement@}\csuse{series@display##1}{##1}%
4324     }{}
4325     \ifstrequal{#4}{familiar}{
4326       \csuse{arrangementX@}\csuse{series@displayX##1}{##1}%
4327     }{}
4328   }%
4329   \ifstreempty{#1}{%
4330     \dolistloop{\@series}%
4331     \ifstreempty{#5}{}%
4332       \docsvlist{#5}%
4333     }
4334   }%
```



```

4335         {%
4336         \docsvlist{#1}%
4337         }%
4338     }
4339     %

```

`\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`.

```

4340 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4341     \def\do##1{
4342         \csgdef{#2@##1}{#3}
4343         \ifstrequal{#4}{critical}{%
4344             \csuse{Xarrangement@\csuse{series@display##1}}{##1}%
4345         }{}
4346         \ifstrequal{#4}{familiar}{%
4347             \csuse{arrangementX@\csuse{series@displayX##1}}{##1}%
4348         }{}%
4349     }%
4350     \ifstreempty{#1}{%
4351         \dolistloop{\@series}%
4352         \ifstreempty{#5}{}{%
4353             \docsvlist{#5}
4354         }
4355     }%
4356     {%
4357         \docsvlist{#1}%
4358     }%
4359 }%
4360 %

```

### 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><pseudoseri>`, like `\Xtwolinesappref`.

```

4361 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
4362   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][1]{%
4363     \setcommand@series{##1}{#1}{##2}[1][#2]%
4364   }%
4365   \ifstrempy{#2}{%
4366     \def\do##1{%
4367       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
4368       [1]{%
4369         \csuse{#1}[##1]{####1}%
4370       }%
4371     }%
4372     \docsvlist{#2}%
4373   }%
4374 }%

```

`\newhooktoggle@series` `\newhooktoggle@series\command names` is a generic command to add new commands for a new toggle hook, like `\Xnumberonlyfirstinline`. The second argument is also used to create commands named `\<hookname><pseudoseri>`, like `\Xtwolinesbutnotmoreappref`.

```

4375 \newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
4376   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
4377     true},usedefault]{%
4378     \settoggle@series{##1}{#1}{##2}[1][#2]%
4379   }%
4380   \ifstrempy{#2}{%
4381     \def\do##1{%
4382       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
4383         \csuse{#1}[##1]%
4384       }%
4385     }%
4386     \docsvlist{#2}%
4387   }%
4388 }%

```

`\newhookcommand@toggle@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 or notes

```

4389 \newcommand{\newhooktoggle@series@reload}[2]{%
4390   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
4391     true},usedefault]{%
4392     \settoggle@series{##1}{#1}{##2}[#2]%
4393   }%
4394 }%

```



`\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.

```

4395 \newcommand{\newhookcommand@series@reload}[2]{%
4396   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][\{%
4397     \setcommand@series{##1}{#1}{##2}[#2]%
4398   }%
4399 }
4400 %

```

### 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.

```

4401 \unless\ifnocritical@
4402   \newhooktoggle@series{Xparindent}
4403   \newhookcommand@series{Xtwolines}[appref]
4404   \newhookcommand@series{Xmorethantwolines}[appref]
4405   \newhooktoggle@series{Xtwolinesbutnotmore}[appref]
4406   \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref]
4407   \newhookcommand@series{Xhangindent}
4408   \newhookcommand@series{Xragged}
4409   \newhookcommand@series{Xhsizetwocol}
4410   \newhookcommand@series{Xhsizethreecol}
4411   \newhookcommand@series{Xcolalign}%
4412   \newhookcommand@series{Xnotenumfont}
4413   \newhookcommand@series{Xbhooknote}
4414   \newhookcommand@series{Xboxsymlinenum}%
4415   \newhookcommand@series{Xsymlinenum}
4416   \newhookcommand@series{Xbeforenumber}
4417   \newhookcommand@series{Xafternumber}
4418   \newhookcommand@series{Xbeforesymlinenum}
4419   \newhookcommand@series{Xaftersymlinenum}
4420   \newhookcommand@series{Xinplaceofnumber}
4421   \newhookcommand@series{Xlemmaseparator}
4422   \newhookcommand@series{Xbeforelemmaseparator}
4423   \newhookcommand@series{Xafterlemmaseparator}
4424   \newhookcommand@series{Xinplaceoflemmaseparator}
4425   \newhookcommand@series{Xtxtbeforenotes}
4426   \newhookcommand@series@reload{Xafterrule}{critical}
4427   \newhooktoggle@series{Xnumberonlyfirstinline}
4428   \newhooktoggle@series{Xnumberonlyfirstintwolines}
4429   \newhooktoggle@series{Xnonumber}
4430   \newhooktoggle@series{Xpstart}
4431   \newhooktoggle@series{Xpstarteverytime}%
4432
4433   \newhooktoggle@series{Xstanza}%
4434   \newhookcommand@series{Xstanzaseparator}%

```



```

4435 \newhooktoggle@series{Xonlypstart}
4436 \newhooktoggle@series{Xnonbreakableafternumber}
4437 \newhooktoggle@series{Xlemmadisablefontselection}
4438 \newhookcommand@series@reload{Xmaxhnotes}{critical}
4439 \newhookcommand@series@reload{Xbeforenotes}{critical}
4440 \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}{critical}%
4441 \newhookcommand@series{Xnotefontsize}
4442
4443
4444 \newhookcommand@series{Xboxlinenum}%
4445 \newhookcommand@series{Xboxlinenumalign}%
4446
4447 \newhookcommand@series{Xboxstartlinenum}%
4448 \newhookcommand@series{Xboxendlinenum}%
4449
4450 \newhookcommand@series{Xafternote}%
4451 \newhookcommand@series{Xparafootsep}
4452
4453 \newhookcommand@series{Xlinerangeseparator}
4454
4455 \fi
4456 %

```

### XXI.3.5 Options for familiar notes

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

```

4457 \unless\ifnofamiliar@
4458 \newhooktoggle@series{parindentX}
4459 \newhookcommand@series{hangindentX}
4460 \newhookcommand@series{raggedX}
4461 \newhookcommand@series{hsizetwocolX}
4462 \newhookcommand@series{hsizethreecolX}
4463 \newhookcommand@series{colalignX}%
4464 \newhookcommand@series{notenumfontX}
4465 \newhookcommand@series{bhooknoteX}
4466 \newhookcommand@series@reload{beforenotesX}{familiar}
4467 \newhookcommand@series@reload{maxhnotesX}{familiar}
4468 \newhooktoggle@series@reload{noteswidthliketwocolumnsX}{familiar}%
4469 \newhookcommand@series@reload{afterruleX}{familiar}
4470 \newhookcommand@series{notefontsizeX}
4471 \newhookcommand@series{afternoteX}
4472 \newhookcommand@series{parafootsepX}
4473 \fi
4474 %

```



### 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.

```

4475 \unless\ifnoend@
4476   \newhookcommand@series{Xendtwolines}[apprefwithpage]
4477   \newhookcommand@series{Xendmoreethantwolines}[apprefwithpage]
4478   \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage]
4479   \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage]
4480   \newhookcommand@series{Xendnotenumfont}
4481   \newhookcommand@series{Xendbhooknote}
4482
4483   \newhookcommand@series{Xendboxlinenum}%
4484   \newhookcommand@series{Xendboxlinenumalign}%
4485
4486   \newhookcommand@series{Xendboxstartlinenum}%
4487   \newhookcommand@series{Xendboxendlinenum}%
4488
4489   \newhookcommand@series{Xendnotefontsize}
4490   \newhooktoggle@series{Xendlemmadisablefontselection}
4491   \newhookcommand@series{Xendlemmaseparator}
4492   \newhookcommand@series{Xendbeforelemmaseparator}
4493   \newhookcommand@series{Xendafterlemmaseparator}
4494   \newhookcommand@series{Xendinplaceoflemmaseparator}
4495
4496   \newhookcommand@series{Xendbeforenumber}%
4497   \newhookcommand@series{Xendafternumber}%
4498
4499   \newhooktoggle@series{Xendparagraph}
4500   \newhookcommand@series{Xendafternote}
4501   \newhookcommand@series{Xendsep}
4502
4503   \newhookcommand@series{Xendinplaceofnumber}%
4504   \newhooktoggle@series{Xendnonumber}%
4505
4506   \newhooktoggle@series{Xendnumberonlyfirstinline}%
4507   \newhooktoggle@series{Xendnumberonlyfirstintwolines}%
4508
4509   \newhookcommand@series{Xendsymlinenum}%
4510   \newhookcommand@series{Xendbeforesymlinenum}%
4511   \newhookcommand@series{Xendaftersymlinenum}%
4512   \newhookcommand@series{Xendboxsymlinenum}%
4513
4514   \newhookcommand@series{Xendbhooklinenum}%
4515   \newhookcommand@series{Xendahooklinenum}%
4516   \newhookcommand@series{Xendbhookinplaceofnumber}%
4517   \newhookcommand@series{Xendahookinplaceofnumber}%
4518
4519   \newhookcommand@series{Xendhangindent}%

```



```

4520 \newhookcommand@series{Xendlinangeseparator}
4521 \fi
4522 %
4523 %

```

## XXI.4 Hooks for a particular footnote

**\newhooktoggle@specific** \newhooktoggle@specific is a generic command to create boolean hook specific to a note.

```

4524 \newcommand{\newhooktoggle@specific}[1]{%
4525   \newtoggle{#1}%
4526   \define@key[mac]{truefootnoteoption}{#1}[]{\global\settoggle{#1}{true}}%
   When enabling footnote option
4527   \define@key[mac]{falsefootnoteoption}{#1}[]{\global\settoggle{#1}{false}
   }}
4528 }
4529 %

```

**\newhookarg@specific** \newhookarg@specific is a generic command to create argumen hook specific to a note.

```

4530 \newcommand{\newhookarg@specific}[1]{%
4531   \define@key[mac]{truefootnoteoption}{#1}{\global\def\linangesep@{##1}}%
   When enabling footnote option
4532   \define@key[mac]{falsefootnoteoption}{#1}{\global\undef\linangesep@}%
   When
4533 }
4534 %

```

And now, we define some hooks specific to a note.

```

4535 \newhooktoggle@specific{fulllines}%
4536 \newhooktoggle@specific{nonum}
4537 \newhooktoggle@specific{nosep}
4538 \newhookarg@specific{linangesep}
4539 %

```

**linangesep@** \linangesep@ is defined by the option linangesep 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.

```

4540 %

```

**\nomk@** \nomk@ toggle is used by reledpar to remove the footnote mark in the text when using \footnoteXmk. Read reledpar handbook.

```

4541 \newtoggle{nomk}%
4542 %

```



## XXI.5 Alias

`\Xnolemmaseparator` `\Xnolemmaseparator[⟨series⟩]` is just an alias for `\Xlemmaseparator[⟨series⟩]{}`.

```
4543 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]{}}
4544 %
```

## 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`

```
4545 \countdef\pageno=0 \pageno=1
4546 \newcommand*{\advancepageno}{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
4547 \else\global\advance\pageno\@ne\fi}
4548
4549 %
```

### XXII.0.2 Extra footnotes output

With luck we might only have to change `\@makecol` and `\@reinserts` of the  $\TeX$ 's kernel. Since `reledmac`, we use `etoolbox`'s patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

`\doxtrafeet` `\doxtrafeet` is the code extending `\@makecol` to cater for the extra `reledmac` feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of  $\TeX$  are first, then familiar familiar footnotes and finally the critical footnotes.

```
4550 \newcommand*{\l@ddoxtrafeet}{%
4551 \IfStrEq{familiar-critical}{\@fnpos}
4552 {\do@feetX\Xdo@feet}%
4553 {%
4554 \IfStrEq{critical-familiar}{\@fnpos}%
4555 {\Xdo@feet\do@feetX}%
4556 {\do@feetX\Xdo@feet}%
4557 }%
4558 }%
4559
4560 %
```

`\Xdo@feet` `\Xdo@feet` is the code extending `\@makecol` to cater for the extra critical feet.



```

4561 \newcommand*{\Xdo@feet}{%
4562   \setbox\@outputbox \vbox{%
4563     \unvbox\@outputbox
4564     \@opXfeet}}
4565 %

```

**\@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.

```

4566 \newcommand\print@Xnotes[1]{%
4567   \csuse{#1footstart}{#1}%
4568   \csuse{#1footgroup}{#1}%
4569 }%
4570 %

```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```

4571 \newcommand*{\@opXfeet}{%
4572   \unless\ifnocritical@%
4573     \gdef\firstXseries@{}%
4574     \def\do##1{%
4575       \ifvoid\csuse{##1footins}\else%
4576         \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4577         \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4578         \print@Xnotes{##1}%
4579       \fi%
4580     }%
4581     \dolistloop{\@series}%
4582   \fi%
4583 }%
4584 %

```

**\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.

```

4585 \newcommand*{\l@ddodoreintrafeet}{%
4586   \IfStrEq{familiar-critical}{\@fnpos}
4587   {\@doreinfeetX\X@doreinfeet}%
4588   {%
4589     \IfStrEq{critical-familiar}{\@fnpos}%
4590     {\X@doreinfeet\@doreinfeetX}%
4591     {\@doreinfeetX\X@doreinfeet}%
4592   }%
4593 }
4594
4595 %

```

**\X@doreinfeet** \X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.



```

4596 \newcommand*{\X@doreinfeet}{%
4597   \unless\ifnocritical@%
4598     \def\do##1{%
4599       \ifvoid\csuse{##1footins}\else%
4600         \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4601       \fi}%
4602   \dolistloop{\@series}
4603   \fi%
4604 }
4605
4606 %

```

`\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
4607 \newcommand\print@notesX[1]{%
4608   \csuse{footstart#1}{#1}%
4609   \csuse{footgroup#1}{#1}%
4610 }%
4611 %

```

We print all the series of notes by looping on them. We check before printing them that they are not voided.

```

4612 \newcommand*{\do@feetX}{%
4613   \unless\ifnofamiliar@%
4614     \gdef\firstseriesX@{}%
4615     \setbox\@outputbox \vbox{%
4616       \unvbox\@outputbox%
4617       \def\do##1{%
4618         \ifvoid\csuse{footins##1}\else%
4619           \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
4620           \global\advance\skip\csuse{footins##1} by\csuse{afterruleX@##1}%
4621           \print@notesX{##1}%
4622         \fi%
4623       }%
4624       \dolistloop{\@series}}%
4625     \fi%
4626   }%
4627
4628 \newcommand{\@doreinfeetX}{%
4629   \unless\ifnofamiliar@%
4630     \def\do##1{%
4631       \ifvoid\csuse{footins##1}\else
4632         \insert%
4633           \csuse{footins##1}
4634           {\unvbox\csuse{footins##1}}%
4635       \fi%
4636     }%
4637     \dolistloop{\@series}%
4638     \fi%

```



```

4639 }%
4640
4641 %

```

### 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.)

```

4642 \@ifclassloaded{memoir}{%
4643 %
    memoir is loaded so we use memoir's built in hooks.
4644 \g@addto@macro{\m@mddoextrafeet}{\l@dddoextrafeet}%
4645 \g@addto@macro{\m@mddodoreinextrafeet}{\l@ddodoreinextrafeet}%
4646 }{%
4647 %

```

memoir has not been loaded, so patch `\@makecol` and `\@reinserts`.

```

4648 \ifpackageloaded{fancyhdr}{%
4649 \patchcmd%
4650   {\latex@makecol}%
4651   {\xdef\@freelist{\@freelist\@midlist}}%
4652   {\xdef\@freelist{\@freelist\@midlist}\l@dddoextrafeet}%
4653   {}%
4654   {\led@error@fail@patch@makecol}%
4655 }{%
4656 \patchcmd%
4657   {\@makecol}%
4658   {\xdef\@freelist{\@freelist\@midlist}}%
4659   {\xdef\@freelist{\@freelist\@midlist}\l@dddoextrafeet}%
4660   {}%
4661   {\led@error@fail@patch@makecol}%
4662 }%
4663
4664 \patchcmd%
4665   {\@reinserts}%
4666   {\ifvbox}%
4667   {\l@ddodoreinextrafeet\ifvbox}%
4668   {}%
4669   {\led@error@fail@patch@reinserts}%
4670 }
4671
4672 %

```

It turns out that `\@doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.



```

4673 \newif\if@led@nofoot
4674
4675 %

```

```

4676 \ifclassloaded{memoir}{%
4677 %

```

If the memoir class is loaded we hook into its modified \@doclearpage.

```

\@mem@extranofeet 78 \g@addto@macro{\@mem@extranofeet}{%%
4679 \def\do#1{%
4680 \unless\ifnocritical@%
4681 \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
4682 \fi%
4683 \unless\ifnofamiliar@%
4684 \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
4685 \fi%
4686 }
4687 \dolistloop{\@series}%
4688 }%
4689 }{%
4690 %

```

As memoir is not loaded we have patch \@doclearpage.

```

\@led@testifnofoot 91 \newcommand*{\@led@testifnofoot}{%
\@doclearpage 92 \@led@nofoottrue%
4693 \ifvoid\footins\else%
4694 \@led@nofootfalse%
4695 \fi%
4696 \def\do##1{%
4697 \unless\ifnocritical@%
4698 \ifvoid\csuse{##1footins}\else%
4699 \@led@nofootfalse%
4700 \fi%
4701 \fi%
4702 \unless\ifnofamiliar@%
4703 \ifvoid\csuse{footins##1}\else%
4704 \@led@nofootfalse%
4705 \fi%
4706 \fi%
4707 }%
4708 \dolistloop{\@series}%
4709 }%
4710
4711 \pretocmd%
4712 {\@doclearpage}%
4713 {\@led@testifnofoot}%
4714 {}%

```



```

4715 {\led@error@fail@patch@@doclearpage}%
4716
4717 \patchcmd%
4718 {\@doclearpage}%
4719 {\ifvoid\footins}%
4720 {\if@led@nofoot}%
4721 {}%
4722 {\led@error@fail@patch@@doclearpage}%
4723
4724 }
4725
4726 %

```

## 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.

```

4727 \list@create{\labelref@list}
4728 %

```

**`\zz@@@`** A convenience macro to zero two labeling counters in one go.

```

4729 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
4730
4731 %

```

**`\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>33</sup>

This version of the original `edmac \label` uses `\@bspack` and `\@espack` to eliminate extra space problems and also use the  $\TeX$  write methods for the `.aux` file.

<sup>33</sup>The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.



Jesse Billett<sup>34</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.

```

4732 \newcommand*{\edlabel}[1]{%
4733   \ifl@dpairing\ifautopar%
4734     \strut%
4735     \fi\fi%
4736     \@bsphack%
4737     \ifledRcol%
4738       \write\linenum@outR{\string\@lab}%
4739       \ifx\labelref@listR\empty%
4740         \xdef\label@refs{\zz@@@}%
4741       \else%
4742         \glp\labelref@listR\to\label@refs%
4743       \fi%
4744       \ifvmode%
4745         \advancelabel@refs%
4746       \fi%
4747   %

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

4748   \protected@write\@auxout{%
4749     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
4750     |{#1}}}%
4751   \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
4752   \else%
4753     \write\linenum@out{\string\@lab}%
4754     \ifx\labelref@list\empty%
4755       \xdef\label@refs{\zz@@@}%
4756     \else%
4757       \glp\labelref@list\to\label@refs%
4758     \fi%
4759     \ifvmode%
4760       \advancelabel@refs%
4761     \fi%
4762   \protected@write\@auxout{%
4763     {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR|{#1}}}%
4764     \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}%
4765     \fi%
4766   \@esphack}%
4767 %

```

`\advancelabel@refs` In cases where `\edlabel` is the first element in a paragraph, we have a problem with  
`\labelrefsparseline` line counts, because line counts change only at the first horizontal box of the paragraph.  
`\labelrefsparsesubline` Hence, we need to test `\edlabel` if it occurs at the start of a paragraph. To do so, we

<sup>34</sup>(jdb43@cam.ac.uk) via the `ctt` thread ‘ledmac cross referencing’, 25 August 2003.



use `\ifvmode`. If the test is true, we must advance by one unit the amount of text we write into the `.aux` file. We do so using `\advance\label@refs` command.

```

4768 \newcounter{line}%
4769 \newcounter{subline}%
4770 \newcommand{\advancelabel@refs}{%
4771   \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
4772   \stepcounter{line}%
4773   \ifsublines@%
4774     \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}
4775   %
4776     \stepcounter{subline}{1}%
4777     \def\label@refs{\theline\thesubline}%
4778   \else%
4779     \def\label@refs{\theline|0}%
4780   \fi%
4781 }
4782 \def\labelrefsparseline#1|#2{#1}
4783 \def\labelrefsparsesubline#1|#2{#2}
4784 %

```

**`\l@make@labels`** The `\l@make@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@make@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.

```

4784 \newcommand*{\l@make@labels}{%
4785 \def\l@make@labels#1|#2|#3|#4|#5{%
4786   \expandafter\ifx\c@name the@label#5\endc@name \relax\else
4787   \led@warn@DuplicateLabel{#5}%
4788   \fi
4789   \expandafter\gdef\c@name the@label#5\endc@name{#1|#2|#3|#4}%
4790   \ignorespaces}
4791 %
4792 %

```

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.

```

4793 \AtBeginDocument{%
4794   \def\l@make@labels#1|#2|#3|#4|#5{%
4795   }
4796 %
4797 %

```

**`\@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



$\TeX$  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`.

`\apptlabel`    `\apptlabel`, if called in `\edtext` will insert automatically both a start and an end label for the current edtext lines.

Label should not be already defined.

Parse the \edtext line numbers.

Use the  $\LaTeX$  standard hack for label.

And now, write the data in the auxiliary file.

```

4824         \ifledRcol%
4825         \protected@write\@auxout{}\%
4826         {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR\{#1:start}}%

```



```

4827 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4828 \protected@write\@auxout{}%
4829 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline
|\l@dparsedendsub|\the\c@pstartR|{#1:end}}}%
4830 \else%
4831 \protected@write\@auxout{}%
4832 {\string\l@dmake@labels\space\l@dparsedstartpage|\
\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstart|{#1:start}}}%
4833 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4834 \protected@write\@auxout{}%
4835 {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline
|\l@dparsedendsub|\the\c@pstart|{#1:end}}}%
4836 \fi%
4837 %

```

Use the  $\TeX$  standard hack for label.

```

4838 \esphack%
4839 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

4840 \else%
4841 \led@warn@AppLabelOutEdtext{#1}%
4842 \fi%
4843 %

```

End of `\applabel`

```

4844 }%
4845 %

```

**`\wrap@edcrossref`** `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

4846 \newrobustcmd{\wrap@edcrossref}[2]{%
4847 \ifdef{\hyperlink}%
4848 {\hyperlink{#1}{#2}}%
4849 {#2}%
4850 }
4851 %

```

**`\edpageref`** If the specified label exists, `\edpageref` gives its page number.

**`\xpageref`** For this reference command, as for the other two, a special version with prefix `x` is provided for use in places where the command is to be scanned as a number, as in `\linenum`. These special versions have two limitations: they do not print error messages if the reference is unknown, and they can't appear as the first label or reference command in the file; you must ensure that a `\edlabel` or a normal reference command appears first, or these `x`-commands will always return zeros.

$\TeX$  already defines a `\pageref`, so changing the name to `\edpageref`.



```

4852 \newcommand*\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@
    dgetref@num{1}{#1}}
4853 \newcommand*\xpageref}[1]{\l@dgetref@num{1}{#1}}
4854
4855 %

```

**\edlineref** If the specified label exists, \lineref gives its line number.

```

\lineref
4856 \newcommand*\edlineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@
    dgetref@num{2}{#1}}}%
4857 \newcommand*\xlineref}[1]{\l@dgetref@num{2}{#1}}%
4858
4859 %

```

**\sublineref** If the specified label exists, \sublineref gives its sub-line number.

```

\sublineref
4860 \newcommand*\sublineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@
    dgetref@num{3}{#1}}}%
4861 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}%
4862
4863 %

```

**\pstartref** If the specified label exists, \pstartref gives its pstart number.

```

\pstartref
4864 \newcommand*\pstartref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@
    dgetref@num{4}{#1}}}%
4865 \newcommand*\xpstartref}[1]{\l@dgetref@num{4}{#1}}%
4866
4867 %

```

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.

```

4868 \newcommand*\l@dref@undefined}[1]{%
4869   \expandafter\ifx\csname the@label#1\endcsname\relax
4870   \led@warn@RefUndefined{#1}%
4871   \fi}
4872
4873 %

```

**\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) or (4) pstart number. (This switching is done by calling \l@dlabel@parse.) The second argument is the label-macro, which because of the \l@lab macro above is defined to be a string of the type 123|456|789.



```

4874 \newcommand*{\l@dgetref@num}[2]{%
4875   \expandafter
4876   \ifx\csname the@label#2\endcsname \relax
4877     000%
4878   \else
4879     \expandafter\expandafter\expandafter
4880     \l@dlabel@parse\csname the@label#2\endcsname|#1%
4881   \fi}
4882
4883 %

```

**\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 three numbers to pick out. (It was earlier given as the first argument of \l@dgetref@num.)

```

4884 \newcommand*{\l@dlabel@parse}{%
4885   \def\l@dlabel@parse#1|#2|#3|#4|#5{%
4886     \ifcase #5%
4887       \or #1%
4888       \or #2%
4889       \or #3%
4890       \or #4%
4891     \fi}
4892 %

```

**\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.

```

4893 \newcommand*{\xxref}[2]{%
4894   {%
4895     \expandafter\ifx\csname the@label#1\endcsname \relax%
4896       \expandafter\let\csname the@@label#1\endcsname\zz@@@%
4897     \else%
4898       \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
4899         {1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
4900       \fi%
4901     \expandafter\ifx\csname the@label#2\endcsname \relax%
4902       \expandafter\let\csname the@@label#2\endcsname\zz@@@%
4903     \else%

```



```

4903 \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
{1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
4904 \fi%
4905 \ifdefined\@Rlineflag%
4906 \StrDel{\csuse{the@@label#1}}{\@Rlineflag}[\@tempa]%
4907 \StrDel{\csuse{the@@label#2}}{\@Rlineflag}[\@tempb]%
4908 \else%
4909 \letcs{\@tempa}{the@@label#1}%
4910 \letcs{\@tempb}{the@@label#2}%
4911 \fi%
4912 \linenum{\@tempa|
4913 \@tempb}}}%
4914
4915 %

```

`\appref` prints a crossref to some lines of the apparatus defined by `\applabel`. It prints the lines as they should be printed in the apparatus.

`\apprefwithpage`

`\@apprefprefixsingle` If `\@apprefprefixsingle` is not empty, it prints it before the line number. If `\@apprefprefixmore` is not empty, it prints it before the line numbers when the first line is not the same as the last line. `\apprefwithpage` prints a crossref to some lines of the apparatus defined by `\applabel`. It always prints the page number, as it should be printed in the end notes. The `\Xtwolinesappref` and `\Xmorethantwolinesappref` are similar to the footnote hooks and `\Xtwolines` `\Xmorethantwolines`.

So, first declare the default value of the hooks for the pseudo-series `appref`. Also declare the internal toggle which are switch by `reledmac`.

```

4916 \xdef\Xtwolines@appref{}%
4917 \xdef\Xmorethantwolines@appref{}%
4918 \newtoggle{Xtwolinesbutnotmore@appref}%
4919 \newtoggle{Xtwolinesonlyinsamepage@appref}%
4920
4921 \xdef\Xendtwolines@apprefwithpage{}%
4922 \xdef\Xendmorethantwolines@apprefwithpage{}%
4923 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
4924 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
4925
4926 %

```

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.

```

4927
4928 \xdef\Xboxstartlinenum@appref{Opt}
4929 \xdef\Xboxendlinenum@appref{Opt}
4930
4931 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
4932 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
4933
4934 %

```



Now, declare the default value of `\@apprefprefixsingle` and `\@apprefprefixmore`, and the commands which defines them

```

4935 \newcommand\@apprefprefixsingle{}%
4936 \newcommand\@apprefprefixmore{}%
4937
4938 \newcommand{\apprefprefixsingle}[1]{%
4939   \gdef\@apprefprefixsingle{#1}%
4940 }
4941
4942 \newcommand{\setapprefprefixmore}[1]{%
4943   \gdef\@apprefprefixmore{#1}%
4944 }
4945
4946 %

```

And now, the main commands: `\appref` and `\apprefwithpage`. These commands call `\printlines` and `\printendlines`. That is why we have previously declared all hooks values tested inside these last commands.

```

4947 \newcommandx{\appref}[2][1,usedefault]{%
4948   \IfStrEq{#1}{fulllines}%
4949     {\toggletrue{fulllines@}}%
4950     {}%
4951   \xdef\@currentseries{appref}%
4952   \ifdefempty{\@apprefprefixmore}%
4953     {\@apprefprefixsingle}%
4954     {%
4955       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
4956       {\@apprefprefixsingle}%
4957       {\@apprefprefixmore}%
4958     }%
4959   \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:start}
4960   }|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4961   \togglefalse{fulllines@}%
4962 }%
4963
4964 % \changes{v1.23.0}{2015/05/18}{Debug \protect\cs{Xendtwolines}, \protect\
4965 % cs{Xendmorethantwolines}, \protect\cs{Xendtwolinesbutnotmore} and \protect\
4966 % cs{Xendtwolinesonlyinsamepage} when using \protect\cs{apprefwithpage}.}
4967 \newcommandx{\apprefwithpage}[2][1,usedefault]{%
4968   \IfStrEq{#1}{fulllines}%
4969     {\toggletrue{fulllines@}}%
4970     {}%
4971   \xdef\@currentseries{apprefwithpage}%
4972   \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
4973   start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4974   \togglefalse{fulllines@}%
4975 }%
4976
4977 %

```



`\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`.

```

4973 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\curname the@label#1\
endcsname{#2}}
4974
4975 %

```

(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. 111 and V.9 p. 82), since `\xxref` makes a call to `\linenum` in order to do its work.)

## XXIV Side notes

Regular `\marginpars` do not work inside numbered text — they do not produce any note but do put an extra unnumbered blank line into the text.

`\@xympar` Changing `\@xympar` a little at least ensures that `\marginpars` in numbered text do not disturb the flow.

```

4976 \pretocmd{\@xympar}%
4977   {\ifnumberedpar@
4978     \led@warn@NoMarginpars
4979     \@esphack
4980   \else}%
4981 {}%
4982 {}%
4983
4984 \apptocmd{\@xympar}%
4985   {\fi}%
4986 {}
4987 {}
4988
4989 %

```

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@dotsidenote@margin` returns the number associated to side note margin:

**left** : 0

**right** : 1



**outer : 2**

**inner : 3**

```

4990 \newcount\sidenote@margin
4991 \newcommand*{\sidenotemargin}[1]{\{%
4992   \l@dgetsidenote@margin{#1}%
4993   \ifnum\@l@tempcntb>\m@ne
4994     \ifledRcol
4995       \global\sidenote@marginR=\@l@tempcntb
4996     \else
4997       \global\sidenote@margin=\@l@tempcntb
4998     \fi
4999   \fi}}
5000 \newcommand*{\l@dgetsidenote@margin}[1]{\%
5001   \def\@tempa{#1}\def\@tempb{left}%
5002   \ifx\@tempa\@tempb
5003     \@l@tempcntb \z@
5004   \else
5005     \def\@tempb{right}%
5006     \ifx\@tempa\@tempb
5007       \@l@tempcntb \@ne
5008     \else
5009       \def\@tempb{outer}%
5010       \ifx\@tempa\@tempb
5011         \@l@tempcntb \tw@
5012       \else
5013         \def\@tempb{inner}%
5014         \ifx\@tempa\@tempb
5015           \@l@tempcntb \thr@@
5016         \else
5017           \led@warn@BadSidenotemargin
5018           \@l@tempcntb \m@ne
5019         \fi
5020       \fi
5021     \fi
5022   \fi}
5023 \sidenotemargin{right}
5024
5025 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
5026 \newbox\l@dlp@rbox
5027 \newbox\l@drp@rbox
5028
5029 %

```

`\ledlsnotewidth` These specify the width of the left/right boxes (initialised to `\marginparwidth`), their  
`\ledrsnotewidth` distance from the text (initialised to `\linenumsep`), and the fonts used.

`\ledlsnotesep`

`\ledrsnotesep`

`\ledlsnotefontsetup`

`\ledrsnotefontsetup`



```

5030 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
5031 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
5032 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
5033 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
5034 \newcommand*\ledlsnotefontsetup{\raggedleft\footnotesize}
5035 \newcommand*\ledrsnotefontsetup{\raggedright\footnotesize}
5036
5037 %

```

`\ledleftnote`, `\ledrightnote`, `\ledinnernote`, `\ledouternote` are the user commands for left, right, inner and outer sidenotes. The two last one are just alias for the two first one, depending of the page number. `\ledsidenote{<text>}` is the command for a moveable sidenote.

```

\ledsidenote
5038 \newcommand*\ledleftnote[1]{\edtext{}{\l@dlsnote{#1}}}
5039 \newcommand*\ledrightnote[1]{\edtext{}{\l@drsnote{#1}}}
5040
5041 \newcommand*\ledinnernote[1]{%
5042   \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5043   \ledleftnote{#1}%
5044   \else%
5045   \ledrightnote{#1}%
5046   \fi%
5047 }
5048
5049 \newcommand*\ledouternote[1]{%
5050   \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
5051   \ledrightnote{#1}%
5052   \else%
5053   \ledleftnote{#1}%
5054   \fi%
5055 }
5056
5057 \newcommand*\ledsidenote[1]{\edtext{}{\l@dcsnote{#1}}}
5058 %

```

`\l@dlsnote` . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```

\l@drsnote
\l@dcsnote
5059 \newif\ifrighnoteup
5060 \righnoteuptrue
5061
5062 \newcommand*\l@dlsnote[1]{%
5063   \begingroup%
5064   \newcommand{\content}{#1}%
5065   \ifnumberedpar@
5066   \ifledRcol%
5067     \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}%

```



```

5068             \to\inserts@listR
5069         \global\advance\insert@countR \@ne%
5070     \else%
5071         \xright@appenditem{\noexpand\vl@dlsnote{\expandonce\content}}}%
5072             \to\inserts@list
5073         \global\advance\insert@count \@ne%
5074     \fi
5075 \fi\ignorespaces\endgroup}
5076
5077 \newcommand*{\l@drsnote}[1]{%
5078     \begingroup%
5079     \newcommand{\content}{#1}%
5080     \ifnumberedpar@
5081         \ifledRcol%
5082             \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}}%
5083                 \to\inserts@listR
5084             \global\advance\insert@countR \@ne%
5085         \else%
5086             \xright@appenditem{\noexpand\vl@drsnote{\expandonce\content}}}%
5087                 \to\inserts@list
5088             \global\advance\insert@count \@ne%
5089         \fi
5090     \fi\ignorespaces\endgroup}
5091
5092 \newcommand*{\l@dcnote}[1]{%
5093     \begingroup%
5094     \newcommand{\content}{#1}%
5095     \ifnumberedpar@
5096         \ifledRcol%
5097             \xright@appenditem{\noexpand\vl@dcnote{\expandonce\content}}}%
5098                 \to\inserts@listR
5099             \global\advance\insert@countR \@ne%
5100         \else%
5101             \xright@appenditem{\noexpand\vl@dcnote{\expandonce\content}}}%
5102                 \to\inserts@list
5103             \global\advance\insert@count \@ne%
5104         \fi
5105     \fi\ignorespaces\endgroup}
5106
5107 %

```

`\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@dcstetext if \ledsidenote is to be put on right.
- \l@dcstetext@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.

```

5108 \newcommand*{\vl@dlsnote}[1]{%
5109   \ifledRcol@%
5110     \@l@tempcntb=\sidenote@marginR%
5111     \ifnum\@l@tempcntb>\@ne%
5112       \advance\@l@tempcntb by\page@numR%
5113     \fi%
5114   \else%
5115     \@l@tempcntb=\sidenote@margin%
5116     \ifnum\@l@tempcntb>\@ne%
5117       \advance\@l@tempcntb by\page@num%
5118     \fi%
5119   \fi%
5120   \ifodd\@l@tempcntb%
5121     \listgadd{\l@dcstetext@l}{#1}%
5122   \else%
5123     \listgadd{\l@dcstetext}{#1}%
5124   \fi
5125 }
5126 \newcommand*{\vl@drsnote}[1]{%
5127   \ifledRcol@%
5128     \@l@tempcntb=\sidenote@marginR%
5129     \ifnum\@l@tempcntb>\@ne%
5130       \advance\@l@tempcntb by\page@numR%
5131     \fi%
5132   \else%
5133     \@l@tempcntb=\sidenote@margin%
5134     \ifnum\@l@tempcntb>\@ne%
5135       \advance\@l@tempcntb by\page@num%
5136     \fi%
5137   \fi%
5138   \ifodd\@l@tempcntb%
5139     \listgadd{\l@dcstetext}{#1}%
5140   \else%
5141     \listgadd{\l@dcstetext@r}{#1}%
5142   \fi%
5143 }
5144 \newcommand*{\vl@dcstetext}[1]{\listgadd{\l@dcstetext}{#1}}
5145
5146 %

```



`\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.

```

5147 \newcommand*\setl@dlp@rbox}[1]{%
5148   {\parindent\z@\hsize=\ledlsnotewidth\ledlsnotefontsetup
5149     \global\setbox\l@dlp@rbox
5150     \ifleftnoteup
5151       =\vbox to\z@{\vss #1}%
5152     \else
5153       =\vbox to 0.70\baselineskip{\strut#1\vss}%
5154     \fi}}
5155 \newcommand*\setl@drp@rbox}[1]{%
5156   {\parindent\z@\hsize=\ledrsnotewidth\ledrsnotefontsetup
5157     \global\setbox\l@drp@rbox
5158     \ifrighnoteup
5159       =\vbox to\z@{\vss#1}%
5160     \else
5161       =\vbox to0.7\baselineskip{\strut#1\vss}%
5162     \fi}}
5163 \newif\ifleftnoteup
5164 \leftnoteuptrue
5165 %

```

`\@sidenotesep` This macro is used to separate sidenotes of the same line.

```

5166 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
5167 \newcommand{\@sidenotesep}{, }
5168 %

```

`\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.

```

5169 \newcommand*\affixside@note){%
5170   \def\sidenotecontent@{}%
5171   \numgdef\itemcount@{0}%
5172   \def\do##1{%
5173     \ifnumequal{\itemcount@}{0}%
5174     {%
5175       \appto\sidenotecontent@{##1}}% Not print not separator before
5176       the 1st note
5177     {\appto\sidenotecontent@{\@sidenotesep ##1}%
5178     }%
5179     \numgdef\itemcount@{\itemcount@+1}%
5180   }%
5181   \dolistloop{\l@dcstotetext}%
5182   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5183 %

```



And we do the same for left and right notes (not movable).

```

5183 \gdef\@templ@d{%
5184 \gdef\@templ@n{\l@dcsnotetext\l@dcsnotetext@l\l@dcsnotetext@r}%
5185 \ifx\@templ@d\@templ@n \else%
5186 \if@twocolumn%
5187 \if@firstcolumn%
5188 \setl@dlp@rbox{##1}{\sidenotecontent@}%
5189 \else%
5190 \setl@drp@rbox{\sidenotecontent@}%
5191 \fi%
5192 \else%
5193 \l@dtempcntb=\sidenote@margin%
5194 \ifnum\l@dtempcntb>\@ne%
5195 \advance\l@dtempcntb by\page@num%
5196 \fi%
5197 \ifodd\l@dtempcntb%
5198 \setl@drp@rbox{\sidenotecontent@}%
5199 \gdef\sidenotecontent@{%
5200 \numgdef{\itemcount@}{0}%
5201 \dolistloop{\l@dcsnotetext@l}%
5202 \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5203 \setl@dlp@rbox{\sidenotecontent@}%
5204 \else%
5205 \setl@dlp@rbox{\sidenotecontent@}%
5206 \gdef\sidenotecontent@{%
5207 \numgdef{\itemcount@}{0}%
5208 \dolistloop{\l@dcsnotetext@r}%
5209 \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5210 \setl@drp@rbox{\sidenotecontent@}%
5211 \fi%
5212 \fi%
5213 \fi%
5214 }
5215 %

```

## 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.

```

5216 \ifnoledgroup\else%
5217 \newcommand*{\l@dfeetbeginmini}{\l@dedbeginmini\l@dfambeginmini}
5218 \newcommand*{\l@dfeetendmini}{%

```



```

5219 \IfStrEq{critical-familiar}{\@mpfnpos}%
5220 {\l@dedendmini\l@dfamendmini}%
5221 {%
5222     \IfStrEq{familiar-critical}{\@mpfnpos}%
5223     {\l@dfamendmini\l@dedendmini}%
5224     {\l@dedendmini\l@dfamendmini}%
5225 }%
5226 }%
5227 %

```

**\l@dedbeginmini** These handle the initiation and closure of critical footnotes in a minipage environment.

**\l@dedendmini**

```

5228 \newcommand*{\l@dedbeginmini}{%
5229     \unless\ifnocritical@%
5230     \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5231     \dolistloop{\@series}%
5232     \fi%
5233 }
5234 \newcommand*{\l@dedendmini}{%
5235     \unless\ifnocritical@%
5236     \ifl@dpairing%
5237         \ifledRcol%
5238             \flush@notesR%
5239         \else%
5240             \flush@notes%
5241         \fi%
5242     \fi
5243     \def\do##1{%
5244         \ifvoid\csuse{mp##1footins}\else%
5245             \ifl@dpairing\ifparledgroup%
5246                 \ifledRcol%
5247                     \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+\
skip\@nameuse{mp##1footins}}%
5248                 \else%
5249                     \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
+\skip\@nameuse{mp##1footins}}%
5250                 \fi%
5251             \fi\fi%
5252             \csuse{mp##1footgroup}{##1}%
5253         \fi}%
5254     \dolistloop{\@series}%
5255     \fi%
5256 }%
5257 %
5258 %

```

**\l@dfambeginmini** These handle the initiation and closure of familiar footnotes in a minipage environment.

**\l@dfamendmini**

```

5259 \newcommand*{\l@dfambeginmini}{%
5260     \unless\ifnofamiliar@%

```



```

5261 \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5262 \dolistloop{\@series}%
5263 \fi%
5264 }%
5265
5266 \newcommand*{\l@dfamendmini}{%
5267 \unless\ifnofamiliar@%
5268 \def\do##1{%
5269 \ifvoid\csuse{mpfootins##1}\else%
5270 \csuse{mpfootgroup##1}{##1}%
5271 \fi}%
5272 \dolistloop{\@series}%
5273 \fi%
5274 }%
5275 %

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

5276 \patchcmd%
5277 {\@iiiminipage}%
5278 {\let\@footnotetext\@mpfootnotetext}%
5279 {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
5280 {}%
5281 {\led@error@fail@patch@\@iiiminipage}%
5282 %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5283 \patchcmd%
5284 {\endminipage}%
5285 {\footnoterule}%
5286 {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5287 {}%
5288 {\led@error@fail@patch@\endminipage}
5289
5290 \patchcmd%
5291 {\endminipage}%
5292 {\@minipagefalse}%
5293 {\l@dfeetendmini\@minipagefalse}%
5294 {}%
5295 {\led@error@fail@patch@\endminipage}
5296
5297 %

```

`\l@dunboxmpfoot` `\l@dunboxmpfoot` insert normal footnotes for `ledgroup`.  
`\l@advance@parledgroup@beforenormalnotes`

```

5298 \newcommand*{\l@dunboxmpfoot}{%
5299 \vskip\skip\@mpfootins
5300 \normalcolor
5301 \footnoterule

```



```

5302 \l@advance@parledgroup@beforenormalnotes
5303 \unvbox\@mpfootins%
5304 }
5305 %

```

When using parallel ledgroup, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5306 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5307 \ifparledgroup
5308 \ifl@dpairing
5309 \ifledRcol
5310 \dimgdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@mpfootins}
5311 \else
5312 \dimgdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+
skip\@mpfootins}
5313 \fi
5314 \fi
5315 \fi
5316 }
5317 %

```

**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.

```

5318 \newenvironment{ledgroup}{%
5319 \resetprevpage@num%
5320 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5321 \let\@footnotetext\@mpfootnotetext
5322 \l@dfeetbeginmini%
5323 }{%
5324 \par
5325 \unskip
5326 \ifvoid\@mpfootins\else
5327 \l@dunboxmpfoot
5328 \fi
5329 \l@dfeetendmini%
5330 }
5331 %
5332 %

```

**ledgroupsize** `\begin{ledgroupsize} [pos] {width}`

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 *width* minipage. The optional *pos* controls the sideways position of numbered text.

```

5333 \newenvironment{ledgroupsize}[2][1]{%
5334 %

```

Set the various text measures.



```

5335 \hspace #2\relax
5336 %% \textwidth #2\relax
5337 %% \columnwidth #2\relax
5338 %

```

Initialize fills for centering.

```

5339 \let\ledllfill\hfil
5340 \let\ledrlfill\hfil
5341 \def\@tempa{#1}\def\@tempb{1}%
5342 %

```

Left adjusted numbered lines

```

5343 \ifx\@tempa\@tempb
5344 \let\ledllfill\relax
5345 \else
5346 \def\@tempb{r}%
5347 \ifx\@tempa\@tempb
5348 %

```

Right adjusted numbered lines

```

5349 \let\ledrlfill\relax
5350 \fi
5351 \fi
5352 %

```

Set up the footnoting.

```

5353 \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5354 \let\@footnotetext\@mpfootnotetext
5355 \l@dfetbeginmini%
5356 }{%
5357 \par
5358 \unskip
5359 \ifvoid\@mpfootins\else
5360 \l@dunboxmpfoot
5361 \fi
5362 \l@dfetendmini%
5363 }
5364
5365 %

```

Close the \ifnoledgroup@\else.

```

5366 \fi%
5367 %

```

**\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`.

```

5368 \newif\ifledgroupnotesL@
5369 \newif\ifledgroupnotesR@
5370 %

```



## XXVI Indexing

Here is some code for indexing using page and line numbers.

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.

```

5371 \AtBeginDocument{%
5372   \unless\ifl@imakeidx%
5373     \ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5374   \fi%
5375   \unless\ifl@indextools%
5376     \ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5377   \fi%
5378 }
5379 %

```

`\pagelinesep` In order to get a correct line number we have to use the label/ref mechanism. These  
`\edindexlab` macros are for that.

```

\c@labidx
5380 \newcommand{\pagelinesep}{-}
5381 \newcommand{\edindexlab}{${&}}
5382 \newcounter{labidx}
5383 \setcounter{labidx}{0}
5384
5385 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5386 \newcommand{\doedindexlabel}{%
5387   \stepcounter{labidx}%
5388   \edlabel{\edindexlab\thelabidx}%
5389 }
5390
5391 %

```

`\thepageline` This macro makes up the page/line number combo from the label/ref.

```

5392 \newcommand{\thepageline}{%
5393   \thepage%
5394   \pagelinesep%
5395   \xlineref{\edindexlab\thelabidx}%
5396 }
5397 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command  
`\theendpageline` is called in critical notes.

```

5398 \newcommand{\thestartpageline}{%
5399   \l@dparsedstartpage%
5400   \pagelinesep%
5401   \l@dparsedstartline%

```



```

5402 }
5403 \newcommand{\theendpageline}{%
5404   \l@dparsedendpage%
5405   \pagelinesep%
5406   \l@dparsedendline%
5407 }
5408 %

```

**\if@edindex@fornote@true** This boolean test is switching at the beginning of each critical note, to allow index referring to this note.

```

5409 \newif\if@edindex@fornote@
5410 %

```

**\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 |.

```

5411 \newcommand{\prepare@edindex@fornote}[1]{%
5412   \l@dp@rsefootspec#1|}%
5413   \@edindex@fornote@true%
5414 }
5415 %

```

**\get@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.

```

5416 \newcommand{\get@edindex@ledinnote@command}{%
5417   \ifxindy%
5418     \gdef\@ledinnote@command{%
5419       ledinnote\thelabidx%
5420     }%
5421     \ifxindyhyperref%
5422       \immediate\write\eledmac@xindy@out{%
5423         (define-attributes ("ledinnote\thelabidx"))^^J
5424         \space\space(markup-locref^^J
5425         \eledmacmarkuplocrefdepth^^J
5426         :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
5427       }{"^^J
5428         :close "}"^^J
5429         :attr "ledinnote\thelabidx"^^J
5430       )
5431     }%
5432   \else%
5433     \immediate\write\eledmac@xindy@out{%
5434       (define-attributes ("ledinnote\thelabidx"))^^J

```



```

5434 \space\space(markup-locref^^J
5435 \eledmacmarkuplocorefdepth^^J
5436 :open "\string\ledinnote{\@index@command}{^^J
5437 :close "}"^^J
5438 :attr "ledinnote\thelabidx"^^J
5439 )
5440 }%
5441 \fi%
5442 %

```

If we do not use xindy option, \@ledinnote@command will produce something like ledinnote{formattingcommand}.

```

5443 \else%
5444 \gdef\@ledinnote@command{%
5445 ledinnote[\edindexlab\thelabidx]{\@index@command}%
5446 }%
5447 \fi%
5448 }
5449 %

```

**\get@index@command** This macro is used to analyse if a text to be indexed has a command after a |.

```

5450 \def\get@index@command#1|#2+{%
5451 \gdef\@index@txt{#1}%
5452 \gdef\@index@command{#2}%
5453 \xdef\@index@parenthesis{%
5454 \IfBeginWith{\@index@command}{(}{%
5455 \StrGobbleLeft{\@index@command}{1}{\@index@command@}%
5456 \global\let\@index@command\@index@command@%
5457 \xdef\@index@parenthesis{(%
5458 }%
5459 \IfBeginWith{\@index@command}{)}{%
5460 \StrGobbleLeft{\@index@command}{1}{\@index@command@}%
5461 \global\let\@index@command\@index@command@%
5462 \xdef\@index@parenthesis{)%
5463 }%
5464 }
5465 %

```

**\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**

```

5466 \newcommandx{\ledinnote}[3][1,usedefault]{%
5467 \ifboolexpr{%
5468 test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
5469 or%
5470 bool {xindyhyperref@}%
5471 }%

```



```

5472   {%
5473   \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
5474   }%
5475   {%
5476   \csuse{#2}{\ledinnotemark{#3}}%
5477   }%
5478 }%
5479 \newcommand{\ledinnotehyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
5480 {#2}}}}%
5481 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
5482 %

```

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
5482 \newcommand{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
index name, #2 = the text
5483 \global\let\old@Rlineflag\@Rlineflag%
5484 \gdef\@Rlineflag{}%
5485 \ifl@imakeidx%
5486 \if@edindex@fornote%
5487 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5488 \get@edindex@ledinnote@command%
5489 \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
}{\thestartpageline}%
5490 \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
}{\theendpageline}%
5491 \else%
5492 \get@edindex@hyperref{#2}%
5493 \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\thepageline}%
5494 \fi%
5495 \else%
5496 \if@edindex@fornote%
5497 \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5498 \get@edindex@ledinnote@command%
5499 \expandafter\protected@write\@indexfile{}%
5500 {\string\indexentry{\@index@txt|(\@ledinnote@command}{\thestartpageline}
5501 }%
5502 \expandafter\protected@write\@indexfile{}%
5503 {\string\indexentry{\@index@txt|)\@ledinnote@command}{\theendpageline}
5504 }%
5505 \else%

```



```

5506 \protected@write\@indexfile{}%
5507 {\string\indexentry{#2}{\thepage\line}
5508 }%
5509 \fi%
5510 \fi%
5511 \endgroup
5512 \global\let\@Rlineflag\old@Rlineflag%
5513 \@esphack%
5514 }
5515 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

5516 \pretocmd{\makeindex}{%
5517 \def\edindex{\@bsphack
5518 \doedindexlabel
5519 \begingroup
5520 \@sanitize
5521 \@wredindex}}{}{}
5522 \newcommand{\edindex}[1]{\@bsphack\@esphack}
5523 %

```

**`\hyperlinkformat`** `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

5524 \newcommand{\hyperlinkformat}[3]{%
5525 \ifstrepty{#1}%
5526 {\hyperlink{#2}{#3}}%
5527 {\csuse{#1}{\hyperlink{#2}{#3}}}%
5528 }}
5529 %

```

**`\hyperlinkR`** `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.

```

5530 \newcommand{\hyperlinkR}[2]{%
5531 \hyperlink{#1}{#2\@Rlineflag}%
5532 }%
5533
5534 %

```

**`\hyperlinkformatR`** `\hyperlinkformatR` command is to be used to create a internal hyperlink, a format and a `\@Rlineflag`, when indexing.

```

5535 \newcommand{\hyperlinkformatR}[3]{%
5536 \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
5537 }%
5538
5539 %

```



`\get@edindex@hyperref` `\@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`).

```
5540 \newcommand{\get@edindex@hyperref}[1]{%
5541 %
```

We have to disable temporary spaces to work through a `xstring` bug (or feature?)

```
5542 \edef\temp@{%
5543 \catcode`\ =9 %space need for catcode
5544 \detokenize{#1}%For active character in unicode
5545 \catcode`\ =10 % space need for catcode
5546 }%
5547 %
```

Now, we define `\@edindex@hyperref` if the hyperindex of `hyperref` is enabled.

```
5548 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
5549 \IfSubStr{\temp@}{|}%
5550 {\get@index@command#1+%
5551 \ifledRcol%
5552 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5553 hyperlinkformatR{\@index@command}%
5554 {\edindexlab\thelabidx}}%
5555 \else%
5556 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5557 hyperlinkformat{\@index@command}%
5558 {\edindexlab\thelabidx}}%
5559 \fi%
5560 }%
5561 {\get@index@command#1+%
5562 \ifledRcol%
5563 \gdef\@edindex@hyperref{hyperlinkR{\edindexlab\thelabidx}}%
5564 \else%
5565 \gdef\@edindex@hyperref{hyperlink{\edindexlab\thelabidx}}%
5566 \fi%
5567 }%
5568 }%
5569 %
```

```
5570 % If we use both xindy and hyperref, first get the \protect\cs{
index@command} command.
5571 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
5572 % \begin{macrocode}
5573 {\ifxindyhyperref%
5574 \IfSubStr{\temp@}{|}%
5575 {\get@index@command#1+%
5576 {\get@index@command#1+%
5577 \gdef\@edindex@hyperref{eledmac\thelabidx}%
5578 %
```

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`.



```

5579 \IfStrEq{\@index@parenthesis}{(}{%
5580 {%
5581 \csxdef{xindy@parenthesis@\@index@txt}{\thelabidx}%
5582 \gdef\@edindex@hyperref{|\@edmac\thelabidx}%
5583 }%
5584 {}%
5585 %

```

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`.

```

5586 \IfStrEq{\@index@parenthesis}{)}{%
5587 {%
5588 \xdef\@edindex@hyperref{|\@edmac\csuse{xindy@parenthesis@\@index@txt}}%
5589 \global\csundef{xindy@parenthesis@\@index@txt}%
5590 }%
5591 %

```

Write in the `.xdy` file the attributes of the location.

```

5592 {%
5593 \immediate\write\@edmac@xindy@out{%
5594 (define-attributes ("eledmac\thelabidx"))^^J
5595 \space\space(markup-locref^^J
5596 \eledmacmarkuplocdepth^^J
5597 :open "\string\hyperlink%
5598 \ifledRcol R\fi%
5599 {\edindexlab\thelabidx}%
5600 {\ifdefempty{\@index@command}%
5601 {}%
5602 {\@backslashchar\@index@command}%
5603 {"^^J
5604 :close "}"^^J
5605 :attr "eledmac\thelabidx"^^J
5606 )
5607 }%
5608 }%
5609 %

```

And now, in any other case.

```

5610 \else%
5611 \gdef\@index@txt{#1}%
5612 \gdef\@edindex@hyperref{}%
5613 \fi%
5614 }%
5615 }
5616 %

```



## XXVII 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`.

### XXVII.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.

```
5617 \def\@hangingsymbol{}
5618 \newcommand*\sethangingsymbol}[1]{%
5619   \gdef\@hangingsymbol{#1}%
5620 }%
5621 \newif\ifinstanza
5622 %
```

`\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.

```
5623 \newif\ifinserthangingsymbol
5624 \newcommand\inserthangingsymbol{%
5625   \ifinserthangingsymbol%
5626     \ifinstanza%
5627       \@hangingsymbol%
5628     \fi%
5629   \fi%
5630 }
5631 %
```

### XXVII.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.

```
5632 \newcommand*\ampersand{\char`\&}
5633
5634 %
```



### XXVII.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.

```
5635 \chardef\body=\catcode`\@
5636 \catcode`\@=11
5637 \chardef\next=\catcode`\&
5638 \catcode`\&=\active
5639
5640 %
```

### XXVII.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.

```
5641 \newcount\stanza@count
5642 \newlength{\stanzaindentbase}
5643 \setlength{\stanzaindentbase}{20pt}
5644
5645 %
```

`\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.

```
5646 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
5647 \newcommand*\setstanzavalues}[2]{\def\@tempa{#2,,|}%
5648   \stanza@count\z@
5649   \def\next{\expandafter\strip@szacnt\@tempa
5650     \ifx\@tempb\empty\let\next\relax\else
5651       \expandafter\mathchardef\csname #1@\number\stanza@count
5652         \@endcsname\@tempb\relax
5653       \advance\stanza@count\@ne\fi\next}%
5654   \next}
5655
5656 %
```

`\setstanzaindents` In the original edmac, `\setstanzavalues{sza}{\langle...⟩}` had to be called to set the in-  
`\setstanzapenalties` dents, and similarly `\setstanzavalues{szp}{\langle...⟩}` 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).

```
5657 \newcommand*\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
5658 \newcommand*\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
```



```
5659 %
5660 %
```

**\managestanza@modulo** Since version 0.13, the `stanzaindentrepetition` 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 `stanzaindentrepetition` counter, the command restarts it.

```
5661 \newcounter{stanzaindentrepetition}
5662 \newcount\stanza@modulo
5663
5664 \newcommand*{\managestanza@modulo}[0]{
5665   \advance\stanza@modulo\@ne
5666   \ifnum\stanza@modulo>\value{stanzaindentrepetition}
5667     \stanza@modulo\@ne
5668   \fi
5669 }
5670 %
```

**\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.

```
5671 \newcommand{\stanzaindent}[1]{%
5672   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
5673   \ignorespaces%
5674 }%
5675 \WithSuffix\newcommand\stanzaindent*[1]{%
5676   \stanzaindent{#1}%
5677   \global\advance\stanza@modulo-\@ne%
5678   \ifnum\stanza@modulo=0%
5679     \global\stanza@modulo=\value{stanzaindentrepetition}%
5680   \fi%
5681   \ignorespaces%
5682 }%
5683 %
```

## XXVII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```
\thestanza84 \newcounter{stanza}
5685 \renewcommand{\thestanza}{%
5686   \textbf{\arabic{stanza}}%
5687 }
5688 %
```



`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```
5689 \newif\ifnumberstanza%
5690 %
```

`\@insertstanzanumber` Now, macro called at the first line of of verse of a stanza.

```
5691 \newcommand{\@insertstanzanumber}[0]{%
5692   \ifnumberstanza%
5693     \ifl@dpairing%
5694       \ifledRcol%
5695         \stanzanumwrapper{\thestanzaR}%
5696       \else%
5697         \stanzanumwrapper{\thestanzaL}%
5698       \fi%
5699     \else%
5700       \stanzanumwrapper{\thestanza}%
5701     \fi%
5702     \setline{1}%
5703   \fi%
5704 }%
5705 %
```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```
5706 \newcommand{\@advancestanzanumber}[0]{%
5707   \ifnumberstanza%
5708     \ifl@dpairing%
5709       \ifledRcol%
5710         \addtocounter{stanzaR}{1}%
5711       \else%
5712         \addtocounter{stanzaL}{1}%
5713       \fi%
5714     \else%
5715       \addtocounter{stanza}{1}%
5716     \fi%
5717   \fi%
5718 }%
5719 %
```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```
5720 \newcommand{\stanzanumwrapper}[1]{%
5721   \flagstanza{#1}%
5722 }%
5723 %
```

## XXVII.6 Stanza number in note

Here, the command called when printing stanza number in notes.



```

5724 \newcommand{\printstanza}[0]{%
5725   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
5726     \ifledRcol{%
5727       \thestanzaR%
5728     \else%
5729       \thestanzaL%
5730     \fi%
5731   }{%
5732     \thestanza%
5733   }%
5734 }
5735 %

```

## XXVII.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.

```

5736 \newcommandx{\stanza@line}[1][1]{
5737   \ifnum\value{stanzaindentrepetition}=0
5738     \parindent=\csname sza@\number\stanza@count
5739       @\endcsname\stanzaindentbase
5740   \else
5741     \parindent=\csname sza@\number\stanza@modulo
5742       @\endcsname\stanzaindentbase
5743     \managestanza@modulo
5744   \fi
5745   \pstart[#1]\stanza@hang\ignorespaces}
5746 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
5747   \hangindent\expandafter
5748     \noexpand\csname sza@0@\endcsname\stanzaindentbase
5749   \hangafter\@ne}
5750 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
5751   \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
5752   \penalty\fi\count@}
5753 %

```

`\@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 `\&`.

```

5754 \xdef\@startstanza[#1]{%
5755   \noexpand\instanzatrue\expandafter
5756   \begingroup%
5757   \catcode`\noexpand\&\active%
5758   \global\stanza@count\@ne\stanza@modulo\@ne
5759   \noexpand\ifnum\expandafter\noexpand
5760   \csname sz@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
5761   \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
5762   \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
5763   \expandafter\noexpand\csname szp@0@\endcsname=\z@
5764   \let\noexpand\sza@penalty\relax\noexpand\fi%
5765   \def\noexpand&{%
5766     \noexpand\newverse[] []}%
5767   \def\noexpand\&{\noexpand\@stopstanza}%
5768   \noexpand\@advancestanzanumber%
5769   \noexpand\stanza@line[#1]\noexpand\@insertstanzanumber%
5770   \let\par\relax\ignorespaces%No paragraph in verses
5771 }
5772
5773 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
5774
5775 \newcommandx{\@stopstanza}[1][1,usedefault]{%
5776   \unskip%
5777   \endlock%
5778   \pend[#1]%
5779   \endgroup%
5780   \instanzafalse%
5781 }
5782
5783 \newcommandx*{\newverse}[2][1,2,usedefault]{%
5784   \unskip%
5785   \endlock\pend[#1]\sza@penalty\global%
5786   \advance\stanza@count\@ne\stanza@line[#2]%
5787 }
5788
5789 %

```

**\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`.

```

5790 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
5791   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
5792
5793 %

```



## XXVII.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.

```
5794 \catcode`\&=\next
5795 \catcode`\@=\body
5796 \setstanzavalues{szp}{0}
5797
5798 %
```

## XXVIII Arrays and tables

### XXVIII.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.

```
5799 \newtoks\@emptytoks
5800
5801 %
```

The rest is from amsmath.

**\l@denvbody** A token register to contain the body.

```
5802 \newtoks\l@denvbody
5803
5804 %
```

**\addtol@denvbody** \addtol@denvbody{arg} adds arg to the token register \l@denvbody.

```
5805 \newcommand{\addtol@denvbody}[1]{%
5806   \global\l@denvbody\expandafter{\the\l@denvbody#1}}
5807
5808 %
```



`\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`.

```

5809 \newcommand{\l@dcollect@body}[1]{%
5810   \l@denvbody{\expandafter#1\expandafter{\the\l@denvbody}}%
5811   \edef\processl@denvbody{\the\l@denvbody\noexpand\end{\@currenenv}}%
5812   \l@denvbody\@emptytoks \def\l@dbegin@stack{b}%
5813   \begingroup
5814     \expandafter\let\csname\@currenenv\endcsname\l@dcollect@@body
5815     \edef\processl@denvbody{\expandafter\noexpand\csname\@currenenv\endcsname}%
5816     \processl@denvbody%
5817   }%
5818 %
5819 %

```

`\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.

```

5820 \def\l@dpush@begins#1\begin#2{%
5821   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
5822 %
5823 %

```

`\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.

```

5824 \def\l@dcollect@@body#1\end#2{%
5825   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
5826     \expandafter\@gobble\l@dbegin@stack}%
5827   \ifx\@empty\l@dbegin@stack
5828     \endgroup
5829     \@checkend{#2}%
5830     \addtol@denvbody{#1}%
5831   \else
5832     \addtol@denvbody{#1\end{#2}}%
5833   \fi
5834   \processl@denvbody % A little tricky! Note the grouping
5835 }
5836 %
5837 %

```



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>

```

## XXVIII.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`.

### XXVIII.2.1 Disabling and restoring commands

`\l@dtabnoexpands` More no expansion for critical and familiar footnotes in tabular environment.

```
5838 \newcommand*\l@dtabnoexpands}{%
5839   \let\rtab=0%
5840   \let\ctab=0%
5841   \let\ltab=0%
5842   \let\rtabtext=0%
5843   \let\ltabtext=0%
5844   \let\ctabtext=0%
5845   \let\edbeforetab=0%
5846   \let\edaftertab=0%
5847   \let\edatleft=0%
5848   \let\edatright=0%
5849   \let\edvertline=0%
5850   \let\edvertdots=0%
5851   \let\edrowfill=0%
5852 }
5853
5854 %
```

`\disable@familiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in `edtabularx` and `edarrayx` environments.

`\restore@familiarnotes`

```
5855 \newcommand{\disable@familiarnotes}{%
5856   \unless\ifnofamiliar%
5857     \def\do##1{%
5858       \csletcs{footnote@@##1}{footnote##1}%
5859       \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
5860         \protected@csxdef{thefnmark##1}{\csuse{thefootnote##1}}%
5861         \csuse{@footnotemark##1}%
5862       }%
5863     }%
5864     \dolistloop{\@series}%
5865   \fi%
5866 }%
5867 \newcommand{\restore@familiarnotes}{%
5868   \unless\ifnofamiliar%
5869     \def\do##1{%
5870       \csletcs{footnote##1}{footnote@@##1}%
5871     }%
5872     \dolistloop{\@series}%
5873   \fi%
}
```



```

5874 }%
5875
5876 %

```

**\disable@sidenotes** The same, for side notes.

**\restore@sidenotes**

```

5877 \newcommand{\disable@sidenotes}{%
5878   \let\@@ledrightnote\ledrightnote%
5879   \let\@@ledleftnote\ledleftnote%
5880   \let\@@ledsidenote\ledsidenote%
5881   \let\ledrightnote\@gobble%
5882   \let\ledleftnote\@gobble%
5883   \let\ledsidenote\@gobble%
5884 }%
5885 \newcommand{\restore@sidenotes}{%
5886   \let\ledrightnote\@@ledrightnote%
5887   \let\ledleftnote\@@ledleftnote%
5888   \let\ledsidenote\@@ledsidenote%
5889 }%
5890 %

```

**\disable@notes** Disable/restore side and familiar notes.

**\restore@notes**

```

5891 \newcommand{\disable@notes}{%
5892   \disable@sidenotes%
5893   \disable@familiarnotes%
5894 }%
5895 \newcommand{\restore@notes}{%
5896   \restore@sidenotes%
5897   \restore@familiarnotes%
5898 }%
5899 %

```

**\EDTEXT** We need to be able to modify the `\edtext` macros and also restore their original definitions.

**\xedtext**

```

5900 \let\EDTEXT=\edtext
5901 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
5902 %

```

**\EDLABEL** We need to be able to modify and restore the `\edlabel` macro.

**\xedlabel**

```

5903 \let\EDLABEL=\edlabel
5904 \newcommand*\xedlabel[1]{\EDLABEL{#1}}
5905 %

```

**\EDINDEX** Macros supporting modification and restoration of `\edindex`.

**\xedindex**

**\nulledindex**



```

5906 \let\EDINDEX=\edindex
5907 \newcommand{\xedindex}{\@bsphack%
5908   \ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
5909 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
5910
5911 %

```

**\@line@num** Macro supporting restoration of \linenum.

```

5912 \let\@line@num=\linenum
5913 %

```

**\l@gobblearg** **\l@gobbleoptarg** [*arg*] {*arg*} replaces these two arguments (first is optional) by \relax.

```

5914 \newcommand*{\l@gobbleoptarg}[2][\relax]%
5915
5916 %

```

**\Relax**<sub>17</sub> \let\Relax=\relax

**\NEXT**<sub>18</sub> \let\NEXT=\next

```

5919
5920 %

```

**\l@dmodforedtext** Modify and restore various macros for when \edtext is used.  
**\l@drestoreforedtext**

```

5921 \newcommand{\l@dmodforedtext}{%
5922   \let\edtext\relax
5923   \def\do##1{\global\csletcs{##1footnote}{\l@gobbleoptarg}}%
5924   \dolistloop{\@series}%
5925   \let\edindex\nulledindex
5926   \let\linenum\@gobble}
5927 \newcommand{\l@drestoreforedtext}{%
5928   \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
5929   \dolistloop{\@series}%
5930   \let\edindex\xedindex}
5931 %

```

**\l@dnullfills** Nullify and restore some column fillers, etc.  
**\l@drestorefills**

```

5932 \newcommand{\l@dnullfills}{%
5933   \def\edlabel##1{%
5934     \def\edrowfill##1##2##3{%
5935     }
5936   \newcommand{\l@drestorefills}{%
5937     \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
5938   }
5939
5940 %

```



`\letsforverteilen` Gathers some lets and other code that is common to the `*verteilen*` macros.

```

5941 \newcommand{\letsforverteilen}{%
5942   \let\edtext\xedtext
5943   \let\edindex\xedindex
5944   \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
5945   \dolistloop{\@series}%
5946   \let\linenum\@line@num
5947   \hilfe skip=\l@dcwidth%
5948   \advance\hilfe skip by -\wd\hilfe box
5949   \def\edlabel##1{\xedlabel{##1}}
5950 %
5951 %

```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for  
`\enablel@dtabfeet` `\edtext`.

```

5952 \newcommand\disablel@dtabfeet{\l@dmodforedtext}%
5953 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
5954 %

```

### XXVIII.2.2 Counters, boxes and lengths

`\l@dampcount` `\l@dampcount` is a counter for the & column dividers and `\l@dcolcount` is a counter  
`\l@dcolcount` for the columns.

```

5955 \newcount\l@dampcount
5956   \l@dampcount=1\relax
5957 \newcount\l@dcolcount
5958   \l@dcolcount=0\relax
5959 %
5960 %

```

`\hilfe box` Some (temporary) helper items.

```

\hilfe skip
5961 \hilfe box \newbox\hilfe box
5962 \hilfe box \newskip\hilfe skip
\hilfe count
5963 \hilfe box \newbox\hilfe box
5964 \hilfe count \newcount\hilfe count
5965 %
5966 %

```

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).

```

5967 \newdimen\dcoli
5968 \newdimen\dcolii
5969 \newdimen\dcoliii
5970 \newdimen\dcoliv
5971 \newdimen\dcolv

```



```

5972 \newdimen\dc colvi
5973 \newdimen\dc colvii
5974 \newdimen\dc colviii
5975 \newdimen\dc colix
5976 \newdimen\dc colx
5977 \newdimen\dc colxi
5978 \newdimen\dc colxii
5979 \newdimen\dc colxiii
5980 \newdimen\dc colxiv
5981 \newdimen\dc colxv
5982 \newdimen\dc colxvi
5983 \newdimen\dc colxvii
5984 \newdimen\dc colxviii
5985 \newdimen\dc colxix
5986 \newdimen\dc colxx
5987 \newdimen\dc colxxi
5988 \newdimen\dc colxxii
5989 \newdimen\dc colxxiii
5990 \newdimen\dc colxxiv
5991 \newdimen\dc colxxv
5992 \newdimen\dc colxxvi
5993 \newdimen\dc colxxvii
5994 \newdimen\dc colxxviii
5995 \newdimen\dc colxxix
5996 \newdimen\dc colxxx
5997 \newdimen\dc colerr % added for error handling
5998
5999 %

```

**\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)

```

6000 \newcommand{\l@dc colwidth}{\ifcase \the\l@dc colcount \dc oli %???
6001 \or \dc oli \or \dc olii \or \dc oliii
6002 \or \dc oliv \or \dc olv \or \dc olvi
6003 \or \dc olvii \or \dc olviii \or \dc olvix \or \dc olx
6004 \or \dc olxi \or \dc olxii \or \dc olxiii
6005 \or \dc olxiv \or \dc olxv \or \dc olxvi
6006 \or \dc olxvii \or \dc olxviii \or \dc olxix \or \dc olxx
6007 \or \dc olxxi \or \dc olxxii \or \dc olxxiii
6008 \or \dc olxxiv \or \dc olxxv \or \dc olxxvi
6009 \or \dc olxxvii \or \dc olxxviii \or \dc olxxix \or \dc olxxx
6010 \else \dc olerr \fi}
6011
6012 %

```

**\step1@dc colcount** This increments the column counter, and issues an error message if it is too large.

```

6013 \newcommand*{\step1@dc colcount}{\advance\l@dc colcount\@ne
6014 \ifnum\l@dc colcount>30\relax

```



```

6015 \led@err@TooManyColumns
6016 \fi}
6017
6018 %

```

**\l@setmaxcolwidth** Sets the column width to the maximum value seen so far.

```

6019 \newcommand{\l@setmaxcolwidth}{%
6020 \ifdim\l@dcwidth < \wd\hilfsbox
6021 \l@dcwidth = \wd\hilfsbox
6022 \else \relax \fi}
6023
6024 %

```

**\measurecell** Measure (recursively) the width required for a math cell.

```

6025 \def\measurecell #1{%
6026 \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
6027 \else\l@dcwidth=0%
6028 \l@dcwidth=0%
6029 \let\NEXT\measurecell%
6030 \fi%
6031 \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6032 \step\l@dcwidth%
6033 \l@setmaxcolwidth%
6034 \let\NEXT\measurecell%
6035 \fi\NEXT}
6036
6037 %

```

**\measuretextcell** Measure (recursively) the width required for a text cell.

```

6038 \def\measuretextcell #1{%
6039 \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
6040 \else\l@dcwidth=0%
6041 \l@dcwidth=0%
6042 \let\NEXT\measuretextcell%
6043 \fi%
6044 \else\setbox\hilfsbox=\hbox{#1}%
6045 \step\l@dcwidth%
6046 \l@setmaxcolwidth%
6047 \let\NEXT\measuretextcell%
6048 \fi\NEXT}
6049
6050 %

```

**\measuremrow** Measure (recursively) the width required for a math row.

```

6051 \def\measuremrow #1{%
6052 \ifx #1\ \let\NEXT\relax%

```



```

6053 \else\measuremcell #1&\\&\\&%
6054 \let\NEXT\measuremrow%
6055 \fi\NEXT}
6056 %

```

**\measuretrow** Measure (recursively) the width required for a text row.

```

6057 \def\measuretrow #1\\{%
6058 \ifx #1&\let\NEXT\relax%
6059 \else\measuretcell #1&\\&\\&%
6060 \let\NEXT\measuretrow%
6061 \fi\NEXT}
6062
6063 %

```

**\edtabcolsep** The length `\edtabcolsep` controls the distance between columns.

```

6064 \newskip\edtabcolsep
6065 \global\edtabcolsep=10pt
6066
6067 %

```

**\variab**<sub>68</sub> `\newcommand{\variab}{\relax}`

```

6069
6070 %

```

**\l@dcheckcols** Check that the number of columns is consistent.

```

6071 \newcommand*{\l@dcheckcols}{%
6072 \ifnum\l@dcolcount=1\relax
6073 \else
6074 \ifnum\l@dampcount=1\relax
6075 \else
6076 \ifnum\l@dcolcount=\l@dampcount\relax
6077 \else
6078 \l@d@err@UnequalColumns
6079 \fi
6080 \fi
6081 \l@dampcount=\l@dcolcount
6082 \fi}
6083
6084 %

```

**\edfilldimen** A length.

```

6085 \newdimen\edfilldimen
6086 \edfilldimen=0pt
6087
6088 %

```



`\c@addcolcount` A counter to hold the number of a column. We use a roman number so that we can grab  
`\theadcolcount` the column dimension from `\dcol`.

```
6089 \newcounter{addcolcount}
6090 \renewcommand{\theadcolcount}{\roman{addcolcount}}
6091 %
```

### XXVIII.2.3 Tabular typesetting

`\setmcellright` Typeset (recursively) cells of display math right justified.

```
6092 \def\setmcellright #1{\def\edlabel##1{}}%
6093 \let\edindex\nulledindex
6094 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
6095 \let\Next\relax%
6096 \else\l@dcolcount=0%
6097 \let\Next=\setmcellright%
6098 \fi%
6099 \else%
6100 \disablel@dtabfeet%
6101 \step1@dcolcount%
6102 \disable@notes%
6103 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6104 \restore@notes%
6105 \letsforverteilen%
6106 \hskip\hilfsskip$\displaystyle{#1}$%
6107 \hskip\edtabcolsep%
6108 \let\Next=\setmcellright%
6109 \fi\Next}
6110 %
6111 %
```

`\settcellright` Typeset (recursively) cells of text right justified.

```
6112 \def\settcellright #1{\def\edlabel##1{}}%
6113 \let\edindex\nulledindex
6114 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
6115 \let\Next\relax%
6116 \else\l@dcolcount=0%
6117 \let\Next=\settcellright%
6118 \fi%
6119 \else%
6120 \disablel@dtabfeet%
6121 \step1@dcolcount%
6122 \disable@notes%
6123 \setbox\hilfsbox=\hbox{#1}%
6124 \restore@notes%
6125 \letsforverteilen%
6126 \hskip\hilfsskip#1%
6127 \hskip\edtabcolsep%
```



```

6128         \let\Next=\settccllright%
6129         \fi\Next}
6130 %

```

**\setmcellleft** Typeset (recursively) cells of display math left justified.

```

6131 \def\setmcellleft #1&{\def\edlabel##1{}%
6132     \let\edindex\nulledindex
6133     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
6134         \else\l@dcolcount=0%
6135         \let\Next=\setmcellleft%
6136         \fi%
6137     \else \disablel@dtabfeet%
6138         \step1@dcolcount%
6139         \disable@notes%
6140         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6141         \restore@notes%
6142         \letsforverteilen%
6143         $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
6144         \let\Next=\setmcellleft%
6145     \fi\Next}
6146 %
6147 %

```

**\settcclleft** Typeset (recursively) cells of text left justified.

```

6148 \def\settcclleft #1&{\def\edlabel##1{}%
6149     \let\edindex\nulledindex
6150     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
6151         \else\l@dcolcount=0%
6152         \let\Next=\settcclleft%
6153         \fi%
6154     \else \disablel@dtabfeet%
6155         \step1@dcolcount%
6156         \disable@notes%
6157         \setbox\hilfsbox=\hbox{#1}%
6158         \restore@notes%
6159         \letsforverteilen%
6160         #1\hskip\hilfsskip\hskip\edtabcolsep%
6161         \let\Next=\settcclleft%
6162     \fi\Next}
6163 %

```

**\setmcellcenter** Typeset (recursively) cells of display math centered.

```

6164 \def\setmcellcenter #1&{\def\edlabel##1{}%
6165     \let\edindex\nulledindex
6166     \ifx #1\\ \ifnum\l@dcolcount=0\let\Next\relax%
6167         \else\l@dcolcount=0%
6168         \let\Next=\setmcellcenter%

```



```

6169         \fi%
6170     \else     \disablel@dtabfeet%
6171               \stepl@dcolcount%
6172               \disable@notes%
6173               \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6174               \restore@notes%
6175               \letsforverteilen%
6176               \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6177               \hskip\edtabcolsep%
6178               \let\Next=\setmcellcenter%
6179     \fi\Next}
6180
6181 %

```

**\settcclcenter** Typeset (recursively) cells of text centered.

```

6182 \def\settcclcenter #1{\def\edlabel##1{%
6183     \let\edindex\nulledindex
6184     \ifx #1\ \ifnum\l@dcolcount=0 \let\Next\relax%
6185         \else\l@dcolcount=0%
6186             \let\Next=\settcclcenter%
6187         \fi%
6188     \else     \disablel@dtabfeet%
6189               \stepl@dcolcount%
6190               \disable@notes%
6191               \setbox\hilfsbox=\hbox{#1}%
6192               \restore@notes%
6193               \letsforverteilen%
6194               \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6195               \hskip\edtabcolsep%
6196               \let\Next=\settcclcenter%
6197     \fi\Next}
6198
6199 %

```

**\NEXT**<sub>00</sub> \let\NEXT=\relax

```

6201
6202 %

```

**\setmrowright** Typeset (recursively) rows of right justified math.

```

6203 \def\setmrowright #1\{%
6204     \ifx #1& \let\NEXT\relax
6205     \else \centerline{\setmcellright #1&\&\&}
6206         \let\NEXT=\setmrowright
6207     \fi\NEXT}
6208 %

```

**\settroright** Typeset (recursively) rows of right justified text.



```

6209 \def\settroright #1\{%
6210   \ifx #1& \let\NEXT\relax
6211   \else \centerline{\settcellright #1&\&\&\&}
6212   \let\NEXT=\settroright
6213   \fi\NEXT}
6214
6215 %

```

**\setmrowleft** Typeset (recursively) rows of left justified math.

```

6216 \def\setmrowleft #1\{%
6217   \ifx #1& \let\NEXT\relax
6218   \else \centerline{\setmcellleft #1&\&\&\&}
6219   \let\NEXT=\setmrowleft
6220   \fi\NEXT}
6221 %

```

**\settrorleft** Typeset (recursively) rows of left justified text.

```

6222 \def\settrorleft #1\{%
6223   \ifx #1& \let\NEXT\relax
6224   \else \centerline{\settcellleft #1&\&\&\&}
6225   \let\NEXT=\settrorleft
6226   \fi\NEXT}
6227
6228 %

```

**\setmrowcenter** Typeset (recursively) rows of centered math.

```

6229 \def\setmrowcenter #1\{%
6230   \ifx #1& \let\NEXT\relax%
6231   \else \centerline{\setmcellcenter #1&\&\&\&}
6232   \let\NEXT=\setmrowcenter
6233   \fi\NEXT}
6234 %

```

**\settrorcenter** Typeset (recursively) rows of centered text.

```

6235 \def\settrorcenter #1\{%
6236   \ifx #1& \let\NEXT\relax
6237   \else \centerline{\settcellcenter #1&\&\&\&}
6238   \let\NEXT=\settrorcenter
6239   \fi\NEXT}
6240
6241 %

```

```

6242 \newcommand{\nullsetzen}{%
6243   \stepl@dc@count%

```



```

6244 \l@dcwidth=0pt%
6245 \ifnum\l@dcwidth=30\let\NEXT\relax%
6246 \l@dcwidth=0\relax
6247 \else\let\NEXT\relax%
6248 \fi\NEXT}
6249
6250 %

```

**\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.

```

6251 \newcommand{\edatleft}[3][\@empty]{%
6252 \ifx#1\@empty
6253 \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
6254 \depth 0pt \right. $\hss}\vfil}
6255 \else
6256 \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
6257 \depth 0pt \right. $\}\vfil}
6258 \fi}
6259 %

```

**\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.

```

6260 \newcommand{\edatright}[3][\@empty]{%
6261 \ifx#1\@empty
6262 \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
6263 \depth 0pt \right#2 $\hss}\vfil}
6264 \else
6265 \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
6266 \depth 0pt \right#2 #1 $\}\vfil}
6267 \fi}
6268
6269 %

```

**\edvertline** `\edvertline{ $\langle len \rangle$ }` vertical line  $\langle len \rangle$  high.

```

6270 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
6271
6272 %

```

**\edvertdots** `\edvertdots{ $\langle len \rangle$ }` vertical dotted line  $\langle len \rangle$  high.

```

6273 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6274 {\cleaders\hbox{$\mathhbox{.}\vbox to 0.5em{ }\vfil}}}}
6275
6276 %

```

**\l@dtabaddcols** `\l@dtabaddcols{ $\langle startcol \rangle$ }{ $\langle endcol \rangle$ }` adds the widths of the columns  $\langle startcol \rangle$  through  $\langle endcol \rangle$  to `\edfilldimen`. It is a  $\text{\LaTeX}$  style reimplementation of the original `\@add@`.



```

6277 \newcommand{\l@dtabaddcols}[2]{%
6278   \l@dccheckstartend{#1}{#2}%
6279   \ifl@dstartendok
6280   \setcounter{addcolcount}{#1}%
6281   \@whilenum \value{addcolcount}<#2\relax \do
6282     {\advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6283      \advance\edfilldimen by \edtabcolsep
6284      \stepcounter{addcolcount}}%
6285   \advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6286   \fi
6287 }
6288
6289 %

```

**\ifl@dstartendok** **\l@dccheckstartend{<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.

```

6290 \newif\ifl@dstartendok
6291 \newcommand{\l@dccheckstartend}[2]{%
6292   \l@dstartendoktrue
6293   \ifnum #1<\@ne
6294     \l@dstartendokfalse
6295     \led@err@LowStartColumn
6296   \fi
6297   \ifnum #2>30\relax
6298     \l@dstartendokfalse
6299     \led@err@HighEndColumn
6300   \fi
6301   \ifnum #1>#2\relax
6302     \l@dstartendokfalse
6303     \led@err@ReverseColumns
6304   \fi
6305 }
6306
6307 %

```

**\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 **\wapunktel** macros.

```

6308 \newcommand*\edrowfill}[3]{%
6309   \l@dtabaddcols{#1}{#2}%
6310   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
6311 \let\@edrowfill@=\edrowfill
6312 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
6313
6314 %

```



`\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`.

```
6315 \newcommand{\leftltab}[1]{%
6316   \hb@xt@z@{\vbox{\edtabindent%
6317     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6318
6319 %
```

`\leftrtab`     `\leftrtab{⟨text⟩}{⟨math⟩}` for `\edbeforetab` in `\rtab`.

```
6320 \newcommand{\leftrtab}[2]{%
6321   #2\hb@xt@z@{\vbox{\edtabindent%
6322     \advance\Hilfsskip by\dcoli%
6323     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6324
6325 %
```

`\leftctab`     `\leftctab{⟨text⟩}{⟨math⟩}` for `\edbeforetab` in `\ctab`.

```
6326 \newcommand{\leftctab}[2]{%
6327   \hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6328     \advance\Hilfsskip by 0.5\dcoli%
6329     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6330     \disablel@dtabfeet$\displaystyle{#2}$}%
6331     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6332     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
6333   #2}
6334
6335 %
```

`\rightctab`     `\rightctab{⟨math⟩}{⟨text⟩}` for `\edaftertab` in `\ctab`.

```
6336 \newcommand{\rightctab}[2]{%
6337   \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6338   \disablel@dtabfeet#2\l@dampcount=\l@dcolcount%
6339   #1\hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6340     \advance\Hilfsskip by 0.5\l@dcolwidth%
6341     \advance\Hilfsskip by -\wd\hilfsbox%
6342     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6343     \disablel@dtabfeet$\displaystyle{#1}$}%
6344     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6345     \advance\Hilfsskip by \edtabcolsep%
6346     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6347   }
6348
6349 %
```



`\rightltab` `\rightltab{<math>}<{<text>}` for `\edaftertab` in `\ltab`.

```

6350 \newcommand{\rightltab}[2]{%
6351     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6352     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6353     #1\hb@xt@{\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6354     \advance\Hilfsskip by\l@dcolwidth%
6355     \advance\Hilfsskip by-\wd\hilfsbox%
6356     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6357     \disablel@dtabfeet$\displaystyle{#1}$}%
6358     \advance\Hilfsskip by-\wd\hilfsbox%
6359     \advance\Hilfsskip by\edtabcolsep%
6360     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6361 }
6362
6363 %

```

`\rightrtab` `\rightrtab{<math>}<{<text>}` for `\edaftertab` in `\rtab`.

```

6364 \newcommand{\rightrtab}[2]{%
6365     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6366     \disablel@dtabfeet#2}%
6367     #1\hb@xt@{\z@{\vbox{\edtabindent%
6368     \advance\Hilfsskip by-\wd\hilfsbox%
6369     \advance\Hilfsskip by\edtabcolsep%
6370     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6371 }
6372
6373 %

```

`\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.

```

6374 \newcommand{\rtab}[1]{%
6375     \l@dnnullfills
6376     \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
6377     \def\edaftertab##1##2{\righttab{##1}{##2}}%
6378     \measurebody{#1}%
6379     \l@dretofills
6380     \variab
6381     \setmrowright #1\\&\\%
6382     \enablel@dtabfeet}
6383
6384 %

```

`\measurebody` `\measurebody{<body>}` measures the array `<body>`.

```

6385 \newcommand{\measurebody}[1]{%
6386     \disablel@dtabfeet%

```



```

6387 \l@dcolcount=0%
6388 \nullsetzen%
6389 \l@dcolcount=0
6390 \measuremrow #1\\&\\%
6391 \global\l@dampcount=1}
6392
6393 %

```

**\rtabtext** `\rtabtext{<body>}` typesets `<body>` as a tabular with the entries right justified.

```

6394 \newcommand{\rtabtext}[1]{%
6395 \l@dnnullfills
6396 \measuretbody{#1}%
6397 \l@drestorefills
6398 \variab
6399 \settroright #1\\&\\%
6400 \enablel@dtabfeet}
6401
6402 %

```

**\measuretbody** `\measuretbody{<body>}` measures the tabular `<body>`.

```

6403 \newcommand{\measuretbody}[1]{%
6404 \disable@notes%
6405 \disablel@dtabfeet%
6406 \l@dcolcount=0%
6407 \nullsetzen%
6408 \l@dcolcount=0
6409 \measuretbody #1\\&\\%
6410 \restore@notes%
6411 \global\l@dampcount=1}
6412
6413 %

```

**\ltab** Array with entries left justified.

```

\edbeforetab
\edaftertab
6414 \newcommand{\ltab}[1]{%
6415 \l@dnnullfills
6416 \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
6417 \def\edaftertab##1##2{\rightltab{##1}{##2}}%
6418 \measuretbody{#1}%
6419 \l@drestorefills
6420 \variab
6421 \setmrowleft #1\\&\\%
6422 \enablel@dtabfeet}
6423
6424 %

```

**\ltabtext** Tabular with entries left justified.



```

6425 \newcommand{\ltabtext}[1]{%
6426   \l@dnnullfills
6427   \measuretbody{#1}%
6428   \l@drestorefills
6429   \variab
6430   \settrorleft #1\\&\\%
6431   \enablel@dtabfeet}
6432
6433 %

```

`\ctab` Array with centered entries.

```

\edbeforetab
\edaftertab
6434 \newcommand{\ctab}[1]{%
6435   \l@dnnullfills
6436   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
6437   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
6438   \measuretbody{#1}%
6439   \l@drestorefills
6440   \variab
6441   \setmrowcenter #1\\&\\%
6442   \enablel@dtabfeet}
6443
6444 %

```

`\ctabtext` Tabular with entries centered.

```

6445 \newcommand{\ctabtext}[1]{%
6446   \l@dnnullfills
6447   \measuretbody{#1}%
6448   \l@drestorefills
6449   \variab
6450   \settrorcenter #1\\&\\%
6451   \enablel@dtabfeet}
6452
6453 %

```

```

\spreadtext54 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
6455   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
6456 %

```

```

\spreadmath57 \newcommand{\spreadmath}[1]{%
6458   \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
6459
6460 %

```

`\HILFSskip` More helpers.

`\Hilfsskip`



```

6461 \newskip\HILFSskip
6462 \newskip\Hilfsskip
6463
6464 %

```

```

\EDTABINDENT \newcommand{\EDTABINDENT}{%
6465
6466   \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
6467   \else\step1@dcolcount%
6468     \advance\Hilfsskip by\l@dcolwidth%
6469     \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
6470     \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
6471     \hilfscount=1\fi%
6472     \let\NEXT=\EDTABINDENT%
6473   \fi\NEXT}%
6474 %

```

`\edtabindent` (was `\tabindent`)

```

6475 \newcommand{\edtabindent}{%
6476   \l@dcolcount=0\relax
6477   \Hilfsskip=0pt%
6478   \hilfscount=1\relax
6479   \EDTABINDENT%
6480   \hilfsskip=\hsize%
6481   \advance\hilfsskip -\Hilfsskip%
6482   \Hilfsskip=0.5\hilfsskip%
6483 }%
6484
6485 %

```

`\EDTAB` (was `\TAB`)

```

6486 \def\EDTAB #1|#2|{%
6487   \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
6488   \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
6489   \advance\tabelskip -\wd\tabhilfbox%
6490   \advance\tabelskip -\wd\tabHilfbox%
6491   \unhbox\tabhilfbox\hskip\tabelskip%
6492   \unhbox\tabHilfbox}%
6493
6494 %

```

`\EDTABtext` (was `\TABtext`)

```

6495 \def\EDTABtext #1|#2|{%
6496   \setbox\tabhilfbox=\hbox{#1}%
6497   \setbox\tabHilfbox=\hbox{#2}%
6498   \advance\tabelskip -\wd\tabhilfbox%
6499   \advance\tabelskip -\wd\tabHilfbox%

```



```

6500 \unhbox\tabhilfbox\hskip\tabelskip%
6501 \unhbox\tabHilfbox}%
6502 %

```

`\tabhilfbox` Further helpers.

```

\tabHilfbox
6503 \newbox\tabhilfbox
6504 \newbox\tabHilfbox
6505
6506 %

```

## XXVIII.2.4 Environments

`edarrayl edarrayc edarrayr` The ‘environment’ forms for `\ltab`, `\ctab` and `\rtab`.

```

6507 \newenvironment{edarrayl}{\l@collect@body\ltab}{\}
6508 \newenvironment{edarrayc}{\l@collect@body\ctab}{\}
6509 \newenvironment{edarrayr}{\l@collect@body\rtab}{\}
6510
6511 %

```

`edtabularl edtabularc edtabularr` The ‘environment’ forms for `\ltabtext`, `\ctabtext` and `\rtabtext`.

```

6512 \newenvironment{edtabularl}{\l@collect@body\ltabtext}{\}
6513 \newenvironment{edtabularc}{\l@collect@body\ctabtext}{\}
6514 \newenvironment{edtabularr}{\l@collect@body\rtabtext}{\}
6515
6516 %

```

## XXIX Quotation's commands

`\initnumbering@quote` This macro, called at the beginning of any numbered section, locally redefines the quotation and quote environments, in order to allow their use inside of numbered sections.

```

\quotation \initnumbering@quote defines quotation environment.
\endquotation
\quote
\endquote
6517 \newcommand{\initnumbering@quote}{
6518 \ifnoquotation@else
6519 \renewcommand{\quotation}{\par\leavevmode%
6520 \parindent=1.5em%
6521 \skipnumbering%
6522 \ifautopar%
6523 \vskip-\parskip%
6524 \else%
6525 \vskip\topsep%
6526 \fi%
6527 \global\leftskip=\leftmargin%

```



```

6528             \global\rightskip=\leftmargin%
6529         }
6530 \renewcommand{\endquotation}{\par%
6531     \global\leftskip=0pt%
6532     \global\rightskip=0pt%
6533     \leavevmode%
6534     \skipnumbering%
6535     \ifautopar%
6536         \vskip-\parskip%
6537     \else%
6538         \vskip\topsep%
6539     \fi%
6540 }
6541 \renewcommand{\quote}{\par\leavevmode%
6542     \parindent=0pt%
6543     \skipnumbering%
6544     \ifautopar%
6545         \vskip-\parskip%
6546     \else%
6547         \vskip\topsep%
6548     \fi%
6549     \global\leftskip=\leftmargin%
6550     \global\rightskip=\leftmargin%
6551 }
6552 \renewcommand{\endquote}{\par%
6553     \global\leftskip=0pt%
6554     \global\rightskip=0pt%
6555     \leavevmode%
6556     \skipnumbering%
6557     \ifautopar%
6558         \vskip-\parskip%
6559     \else%
6560         \vskip\topsep%
6561     \fi%
6562 }
6563 \fi
6564 }
6565 %

```

## XXX Section's title commands

### XXX.1 Commands to disable some feature

`\ledsectnotoc` The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```

6566 \newcommand{\ledsectnotoc}{\let\addcontentsline\@gobblethree}
6567 %

```



`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```

6568 \newcommand{\ledsectnomark}{%
6569   \let\chaptermark@gobble%
6570   \let\sectionmark@gobble%
6571   \let\subsectionmark@gobble%
6572 }
6573 %

```

## XXX.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  $\text{\TeX}$  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  $\text{\TeX}$  sectioning command.
4. This last command is called when the `pstart` is effectively printed.

## XXX.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).

```

6574 \catcode`\#=12
6575 \notbool{@noeled@sec}{%
6576 %

```



`\beforeeledchapter` For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by `eledmac`. Users have to use `\beforeeledchapter`.

```

6577 \ifl@dmemoir
6578   \newcommand\beforeeledchapter{%
6579     \clearforchapter%
6580   }
6581 \else
6582   \newcommand\beforeeledchapter{%
6583     \if@openright%
6584       \cleardoublepage%
6585     \else%
6586       \clearpage%
6587     \fi%
6588   }
6589 \fi
6590 %

```

## XXX.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.

```

6591 \newif\if@eled@sectioning
6592 %

```

`\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.

```

6593 \def\print@rightmargin@eledsection{%
6594   \if@eled@sectioning%
6595     \begingroup%
6596     \if@RTL%
6597       \let\llap\rlap%
6598       \let\leftlinenum\rightlinenum%
6599       \let\leftlinenumR\rightlinenumR%
6600       \let\l@drd@ta\l@dld@ta%
6601       \let\l@drsn@te\l@dlsn@te%
6602     \fi%
6603     \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
6604   \endgroup%
6605 \fi%
6606 }%
6607
6608 \def\print@leftmargin@eledsection{%
6609   \if@eled@sectioning%
6610     \leavevmode%
6611     \begingroup%

```



```

6612 \if@RTL%
6613     \let\rlap\llap%
6614     \let\rightlinenum\leftlinenum%
6615     \let\rightlinenumR\leftlinenumR%
6616     \let\l@dld@ta\l@dld@ta%
6617     \let\l@dlsn@te\l@dlsn@te%
6618 \fi%
6619 \l@dld@ta\csuse{LR}{\l@dlsn@te}%
6620 \endgroup%
6621 \fi%
6622 }%
6623
6624 %

```

### XXX.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.
- `\@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.

```

6625 \AtBeginDocument{%
6626 \patchcmd{\chapter}{\clearforchapter}{%
6627     \if@eled@sectioning\else%
6628     \ifl@printingpages\else%
6629     \clearforchapter%
6630 \fi%
6631 \fi%
6632 }
6633 {}
6634 {}
6635
6636
6637 \pretocmd{\M@sect}
6638 { \let\old@edtext=\edtext%
6639   \let\edtext=\dummy@edtext@showlemma%
6640 }
6641 {}
6642 {}
6643
6644 \apptocmd{\M@sect}
6645 { \let\edtext=\old@edtext}
6646 {}
6647 {}

```



```

6648 \patchcmd{\M@sect}
6649 { #9}
6650 { #9%
6651 \print@rightmargin@eledsection%
6652 }
6653 {}
6654 {}
6655 {}
6656
6657 \patchcmd{\M@sect}
6658 {\hskip #3\relax}
6659 {\hskip #3\relax%
6660 \print@leftmargin@eledsection%
6661 }
6662 {}
6663 {}
6664
6665 \patchcmd{\@mem@old@ssect}
6666 {#5}
6667 {#5%
6668 \print@leftmargin@eledsection%
6669 }
6670 {}
6671 {}
6672
6673 \patchcmd{\@mem@old@ssect}
6674 {\hskip #1}
6675 {\hskip #1%
6676 \print@rightmargin@eledsection%
6677 }
6678 {}
6679 {}
6680
6681 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
6682 \if@eled@sectioning\else%
6683 \ifl@dprintingpages\else%
6684 \if@openright\cleardoublepage\else\clearpage\fi%No clearpage inside a
6685 \Pages: will keep critical notes from printing on the title page. Here for
6686 classical classes
6687 \fi%
6688 \fi%
6689 }%
6690 {}%
6691 {}%
6692
6693 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\
6694 fi}{%
6695 \if@eled@sectioning\else%
6696 \ifl@dprintingpages\else%
6697 \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%
\fi%
}
{}
{}
{}
\patchcmd{\@makechapterhead}
{#1}
{\print@leftmargin@eledsection%
#1%
\print@rightmargin@eledsection%
}
{}
{}
\patchcmd{\@makechapterhead}% For BIDI
{\if@RTL\raggedleft\else\raggedright\fi}%
{\if@eled@sectioning\else%
\if@RTL\raggedleft\else\raggedright\fi%
\fi%
}%
{}%
{}%
\patchcmd{\@makeschapterhead}
{#1}
{\print@leftmargin@eledsection%
#1%
\print@rightmargin@eledsection%
}
{}
{}
\pretocmd{\@sect}
{\let\old@edtext=\edtext
\let\edtext=\dummy@edtext@showlemma%
}
{}
{}
\apptocmd{\@sect}
{\let\edtext=\old@edtext}
{}
{}
\pretocmd{\@ssect}
{\let\old@edtext=\edtext%
\let\edtext=\dummy@edtext@showlemma%

```



```

6743 }
6744 {}
6745 {}
6746
6747 \apptocmd{\@ssect}
6748 {\let\edtext=\old@edtext}
6749 {}
6750 {}
6751
6752 %

```

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.

```

6753 \@ifpackageloaded{nameref}{
6754
6755   \patchcmd{\NR@sect}
6756     {#8}
6757     {#8%
6758       \print@rightmargin@eledsection%
6759     }
6760     {}
6761     {}
6762
6763   \patchcmd{\NR@sect}
6764     {\hskip #3\relax}
6765     {\hskip #3\relax%
6766       \print@leftmargin@eledsection%
6767     }
6768     {}
6769     {}
6770
6771   \patchcmd{\NR@ssect}
6772     {#5}
6773     {#5%
6774       \print@rightmargin@eledsection%
6775     }
6776     {}
6777     {}
6778
6779   \patchcmd{\NR@ssect}
6780     {\hskip #1}
6781     {\hskip #1%
6782       \print@leftmargin@eledsection%
6783     }
6784     {}
6785     {}
6786   }%
6787   {
6788     \patchcmd{\@sect}
6789       {#8}

```



```

6790     {#8%
6791     \print@rightmargin@eledsection%
6792     }
6793     {}
6794     {}
6795
6796     \patchcmd{\@sect}
6797     {\hskip #3\relax}
6798     {\hskip #3\relax%
6799     \print@leftmargin@eledsection%
6800     }
6801     {}
6802     {}
6803
6804     \patchcmd{\@ssect}
6805     {#5}
6806     {#5%
6807     \print@rightmargin@eledsection%
6808     }
6809     {}
6810     {}
6811
6812     \patchcmd{\@ssect}
6813     {\hskip #1}
6814     {\hskip #1%
6815     \print@leftmargin@eledsection%
6816     }
6817     {}
6818     {}
6819     }%
6820 }
6821 %

```

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.

```

6822 {}}%
6823 \protect\catcode`\#=6 %Space NEEDS by \catcode
6824 \notbool{@noeled@sec}{%
6825 %

```

### XXX.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.

```

6826 \newwrite\eled@sectioning@out
6827 %

```



\eledchapter And now, the user sectioning commands, which write to the file, and also add content  
 \eledsection as a “normal” line.

```

\eledsubsection
\eledsubsubsection
\eledchapter*
\eledsection*
\eledsubsection*
\eledsubsubsection*

6828 \newcommand{\eledchapter}[2] [] {%
6829   #2%
6830   \ifledRcol%
6831     \immediate\write\eled@sectioningR@out{%
6832       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6833     }%
6834   \else%
6835     \immediate\write\eled@sectioning@out{%
6836       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6837     }%
6838   \fi%
6839 }
6840
6841 \newcommand{\eledsection}[2] [] {%
6842   #2%
6843   \ifledRcol%
6844     \immediate\write\eled@sectioningR@out{%
6845       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6846     }%
6847   \else%
6848     \immediate\write\eled@sectioning@out{%
6849       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6850     }%
6851   \fi%
6852 }
6853
6854 \newcommand{\eledsubsection}[2] [] {%
6855   #2%
6856   \ifledRcol%
6857     \immediate\write\eled@sectioningR@out{%
6858       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6859     }%
6860   \else%
6861     \immediate\write\eled@sectioning@out{%
6862       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6863     }%
6864   \fi%
6865 }
6866 \newcommand{\eledsubsubsection}[2] [] {%
6867   #2%
6868   \ifledRcol%
6869     \immediate\write\eled@sectioningR@out{%
6870       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
6871     }-}{R}
6872   \else%

```



```

6873 \immediate\write\eled@sectioning@out{%
6874 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
6875 }{}{}
6876 }%
6877 \fi%
6878 }
6879
6880 \WithSuffix\newcommand\eledchapter*[2][]{%
6881 #2%
6882 \ifledRcol%
6883 \immediate\write\eled@sectioningR@out{%
6884 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
6885 }%
6886 \else%
6887 \immediate\write\eled@sectioning@out{%
6888 \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dumpstartsL}{*}{R}
6889 }%
6890 \fi%
6891 }
6892
6893 \WithSuffix\newcommand\eledsection*[2][]{%
6894 #2%
6895 \ifledRcol%
6896 \immediate\write\eled@sectioningR@out{%
6897 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
6898 }%
6899 \else%
6900 \immediate\write\eled@sectioning@out{%
6901 \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dumpstartsL}{*}{R}
6902 }%
6903 \fi%
6904 }
6905
6906 \WithSuffix\newcommand\eledsubsection*[2][]{%
6907 #2%
6908 \ifledRcol%
6909 \immediate\write\eled@sectioningR@out{%
6910 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR}{*}{R}
6911 R}
6912 }%
6913 \else%
6914 \immediate\write\eled@sectioning@out{%
6915 \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
6916 }{*}{R}
6917 }%
6918 \fi%
6919 }
6920
6921 \WithSuffix\newcommand\eledsubsubsection*[2][]{%

```



```

6920 #2%
6921 \ifledRcol%
6922 \immediate\write\eled@sectioningR@out{%
6923 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
6924 }{*}{R}
6925 }%
6926 \else%
6927 \immediate\write\eled@sectioning@out{%
6928 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
6929 }{*}{L}
6930 }%
6931 \fi%

```

### XXX.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.

```

6932 \def\eled@chapter#1#2#3#4#5{%
6933 \ifstrempy{#4}%
6934 {%
6935 \ifstrempy{#1}%
6936 {%
6937 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%
6938 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%
6939 }%Need for \pairs, because of using parbox.
6940 {%
6941 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter[#1]{#2}}%
6942 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{}\
chaptermark{#2}}%Need for \pairs, because of using parbox.
6943 }%
6944 }%
6945 {%
6946 \ifstrempy{#1}%
6947 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%

```



```

6948     {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
6949     }%
6950     \listcsadd{eled@sections#5@@}{#3}%
6951     }
6952     \def\eled@section#1#2#3#4#5{%
6953     \ifstrempy{#4}%
6954     {\ifstrempy{#1}%
6955     {%
6956     \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
6957     \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\
sectionmark{#2}}}%Need for \pairs, because of using parbox.
6958     }%
6959     {%
6960     \global\csdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
6961     \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\
sectionmark{#1}}}%Need for \pairs, because of using parbox.
6962     }%
6963     }%
6964     {\ifstrempy{#1}%
6965     {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
6966     {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
6967     }
6968     \listcsadd{eled@sections#5@@}{#3}%
6969     }
6970     \def\eled@subsection#1#2#3#4#5{%
6971     \ifstrempy{#4}%
6972     {\ifstrempy{#1}%
6973     {%
6974     \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
6975     \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)
6976     }%
6977     {%
6978     \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
6979     \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)
6980     }%
6981     }%
6982     {\ifstrempy{#1}%
6983     {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
6984     {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
6985     }
6986     \listcsadd{eled@sections#5@@}{#3}%
6987     }
6988     \def\eled@subsubsection#1#2#3#4#5{%

```



```

6989 \ifstrempy{#4}%
6990     {\ifstrempy{#1}%
6991       {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%
6992       {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#1}{#2}}}%
6993     }%
6994     {\ifstrempy{#1}%
6995       {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
6996       {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
6997     }
6998 \listcsgadd{eled@sections#5@@}{#3}%
6999 }
7000
7001 %

```

End of the conditional test about noeledsec option.

```

7002 }{}
7003 %

```

## XXXI 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.

```

7004 \def\normal@page@break{}
7005 %

```

**\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.

```

7006 \def\l@prev@pb{}
7007 \def\l@prev@nopb{}
7008 %

```

**\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.



```

7009 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
7010 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
7011 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}
7012 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
7013 %

```

**\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.

```

7014 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
7015 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
7016 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
7017 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
7018 %

```

**\ledpbsetting** The **\ledpbsetting** macro only changes the value of **\led@pb@macro**, for which the **\led@pb@setting** default value is before.

```

7019 \def\led@pb@setting{before}
7020 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
7021 %

```

**\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**.

```

7022 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
7023 \newcommand{\led@check@nopb}{%
7024   \IfStrEq{\led@pb@setting}{before}{%
7025     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
7026       {\numdef{\abs@prevline}{\the\absline@num-1}%
7027       \xifinlist{\abs@prevline}{\normal@page@break}%
7028       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7029       {}}%
7030     {}}%
7031   }%
7032 }%
7033 \IfStrEq{\led@pb@setting}{after}{%
7034   \xifinlist{\the\absline@num}{\l@prev@nopb}{%
7035     \xifinlist{\the\absline@num}{\normal@page@break}%
7036     {\nopagebreak[4]\enlargethispage{\baselineskip}}%
7037     {}}%
7038 }%
7039   {}}%
7040   {}}%
7041 {}%
7042 }
7043 %

```



## XXXII 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.

```

7044 \newcommand{\check@pb@in@verse}{%
7045   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
       enabling page breaks in verse control, while on a hanging verse.
7046   \ifnum\page@num=\last@page@num\else%If we have change page
7047   \IfStrEq{\led@pb@setting}{before}{%
7048     \numgdef{\abs@line@verse}{\the\absline@num-1}%
7049     \ledpbnum{\abs@line@verse}%
7050   }{}%
7051   \IfStrEq{\led@pb@setting}{after}{%
7052     \numgdef{\abs@line@verse}{\the\absline@num-1}%
7053     \lednopbnum{\abs@line@verse}%
7054   }{}%
7055   \fi%
7056   \fi\fi\fi%
7057 }
7058 %

```

## XXXIII Compatibility with eledmac

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

```

7059 \ifeledmaccompat@%
7060
7061 \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
7062 \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
7063 \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
7064 \newcommand{\footthreecolX}[1]{\arrangementX[#1]{threecol}}%
7065
7066 \unless\ifnocritical@
7067   \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
7068   \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
7069   \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
7070   \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%

```



```

7071 \let\hsizetwocol\Xhsizetwocol
7072 \let\hsizethreecol\Xhsizethreecol
7073 \let\bhookXnote\Xbhooknote
7074 \let\boxsymlinenum\Xboxsymlinenum
7075 \let\symlinenum\Xsymlinenum
7076 \let\beforenumberinfootnote\Xbeforenumber
7077 \let\afternumberinfootnote\Xafternumber
7078 \let\beforeXsymlinenum\Xbeforesymlinenum
7079 \let\afterXsymlinenum\Xaftersymlinenum
7080 \let\inplaceofnumber\Xinplaceofnumber
7081 \let\Xlemmaseparator\lemmaseparator
7082 \let\afterlemmaseparator\Xafterlemmaseparator
7083 \let\beforelemmaseparator\Xbeforelemmaseparator
7084 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
7085 \let\txbeforeXnotes\Xtxbeforenotes
7086 \let\afterXrule\Xafterrule
7087 \let\numberonlyfirstinline\Xnumberonlyfirstinline
7088 \let\numberonlyfirstintwoline\Xnumberonlyfirstintwoline
7089 \let\nonumberinfootnote\Xnonumberinfootnote
7090 \let\pstartinfootnote\Xpstart
7091 \let\pstartinfootnoteeverytime\Xpstarteverytime
7092 \let\onlyXpstart\Xonlypstart
7093 \let\Xnonumberinfootnote\Xnonumber
7094 \let\nonbreakableafternumber\Xnonbreakableafternumber
7095 \let\maxhXnotes\Xmaxhnotes
7096 \let\beforeXnotes\Xbeforenotes
7097 \let\boxlinenum\Xboxlinenum
7098 \let\boxlinenumalign\Xboxlinenumalign
7099 \let\boxstartlinenum\Xboxstartlinenum
7100 \let\boxendlinenum\Xboxendlinenum
7101 \let\twoline\Xtwoline
7102 \let\morethantwoline\Xmorethantwoline
7103 \let\twolinebutnotmore\Xtwolinebutnotmore
7104 \let\twolineonlyinsamepage\Xtwolineonlyinsamepage
7105 \fi
7106
7107 \unless\ifnofamiliar@
7108 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
7109 \fi
7110 \newcommandx{\parafootsep}[2][1,usedefault]{%
7111   \Xparafootsep[#1]{#2}%
7112   \parafootsepX[#1]{#2}
7113 }%
7114
7115 \newcommandx{\afternote}[2][1,usedefault]{%
7116   \Xafternote[#1]{#2}%
7117   \afternoteX[#1]{#2}%
7118 }%
7119
7120 \unless\ifnoend@

```



```

7121 \let\XendXtwolines\Xendtwolines
7122 \let\XendXmorethantwolines\Xendmorethantwolines
7123 \let\bhookXendnote\Xendbhooknote
7124 \let\boxXendlinenum\Xendboxlinenum%
7125 \let\boxXendlinenumalign\Xendboxlinenumalign%
7126 \let\boxXendstartlinenum\Xendboxstartlinenum%
7127 \let\boxXendendlinenum\Xendboxendlinenum%
7128 \let\XendXlemmaseparator\Xendlemmaseparator
7129 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
7130 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
7131 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
7132 \fi
7133
7134 \AtBeginDocument{%
7135   \ifdef\lineref{}\let\lineref\edlineref}%
7136 }%
7137
7138
7139 \fi%
7140 %

```

</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>35</sup> to `\critext`; use the `\edtext` macro instead. However, if you do use `\critext` (the new name for `\text`), the following is a reminder.

`\critext`

Within numbered paragraphs, footnotes and endnotes are generated by forms of the `\critext` macro:

```
\critext{⟨lemma⟩}{⟨commands⟩}/
```

The `⟨lemma⟩` argument is the lemma in the main text: `\critext` both prints this as part of the text, and makes it available to the `⟨commands⟩` you specify to generate notes. The `/` at the end terminates the command; it is part of the macro's definition so that spaces after the macro will be treated as significant.

For example:

I saw my friend <code>\critext{Smith}</code>	1 I saw my friend
<code>\Afootnote{Jones C, D.}/</code>	2 Smith on Tuesday.
on Tuesday.	<u>2 Smith</u> ] Jones C, D.

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

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

<code>\critext{I saw my friend</code>	1 I saw my friend
<code>\critext{Smith}{\Afootnote{Jones</code>	2 Smith on Tuesday.
<code>C, D.}/ on Tuesday.}</code>	<u>2 Smith</u> ] Jones C, D.
<code>\Bfootnote{The date was</code>	1-2 I saw my friend
<code>July 16, 1954.}</code>	Smith on Tuesday.] The
<code>/</code>	date was July 16, 1954.

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

The second argument of the `\critext` macro, `⟨commands⟩`, is the same as the second argument to the `\edtext` macro.

It is possible to define aliases for `\critext`, which can be easier to type. You can make a single character substitute for `\critext` by saying this:

<sup>35</sup>A name like `\text` is likely to be defined by other  $\text{\LaTeX}$  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. 103 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. 28), 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. 22). Otherwise please add a new ticket on Github to request a new function for doing this.<sup>36</sup>

If for some reason you do not want to make the modifications to use the new functions of `eledmac`, you can continue using your own `\XXXXXfmt` command, but you must replace:

```
\renewcommand*{XXXXfmt}[3]
```

with

```
\renewcommandx*{XXXXfmt}[4][4=Z]
```

---

<sup>36</sup><https://github.com/maieul/ledmac/issues>



If you do not make that, you will get a spurious [X], where X is series letter.

If you used a `\protect` command inside a `\footnote` command inside a numbered section, you must change the `\protect` to `\noexpand`. Otherwise the command after the `\protect` will be discarded.

### Appendix A.3 Migration to eledmac 1.5.1

The version 1.5.1 corrects a bug in `stanzaindentsrepetition` (cf. 8.3 p. 40). 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. 16). 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 14.2 p. 53.



## Appendix A.5 Migration to eledmac 17.1

This version changes the default setting of `\Xpstart`. Henceforth, `pstart` numbers will be printed in footnotes within the section of text where you have called `\numberpstarttrue`.

We do not see any reason to print them in the other sections. However, if you want to print the `pstart` numbers in all of the footnotes, whatever the section, without having to use `\numberpstarttrue`, you can use `\Xpstarteverytime`.

## Appendix A.6 Migration to eledmac 1.21.0

### Appendix A.6.1 `\Xledsetnormalparstuff` and `\ledsetnormalparstuffX`

The `\ledsetnormalparstuff` has been split into two different commands:

- `\Xledsetnormalparstuff` for critical notes;
- `\ledsetnormalparstuffX` for familiar notes.

Both commands can take an optional argument which is the series letter. If you have redefined `\ledsetnormalparstuff` or any of the commands which call them, you must change them accordingly.

### Appendix A.6.2 Endnotes

In any case, delete the `.end` file before the next run.

The previous version of Eledmac had a bug: there were two spaces between the starting page number and the starting line number, but only one space between the ending page number and the ending line number.

As a matter of fact, a spurious space was added after the first `\printnpnum`. This spurious space has been deleted. However, if you want to keep the previous spurious space, you may load the package with the `oldprintnpnumspace` option.

If you have redefined `\endprint`, you must:

- Contact us and ask for the feature that required your hack, in order to avoid such a hack in the future.
- Use the new fifth argument.
- Add `\xdef\@currentseries{#4}` at the beginning of your own command.

## Appendix A.7 Migration to eledmac 1.22.0

The `\ledinnote` command now takes a first optional argument, which is the label for the hyperreference. If you have redefined it, change your redefinition, and check whether you can avoid this redefinition by only redefining `\ledinnotemark`.

## Appendix A.8 Migration to eledmac 1.23.0

You must delete the numbered auxiliary files before compiling with the new version of eledmac.



## Appendix A.9 Migration from eledmac to reledmac

There are many changes in reledmac which require the user to make modifications.

### Appendix A.9.1 Risk of ‘no room for a new’

The risk to obtain a ‘no room for a new something’ error is greater in reledmac than it is in eledmac. See 17.2 p. 56 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 \renewcommand</i>	<i>Replaced with</i>
\@led@extranofeet	\newseries
\apprefprefixmore	\setapprefprefixmore
\apprefprefixsingle	\setapprefprefixsingle
\endstanzaextra	Optional argument of \&
\hangingsymbol	\sethangingsymbol
\ledfootinsdim	\Xmaxhnotes and \maxhnotesX
\parafootftmsep	\Xparafootsep and \parafootsepX
\notenumfont	\Xnotenumfont, \Xendnotenumfont and \notenumfontX
\notefontsetup	\Xnotefontsize, \Xendnotefontsize and \notefontsizeX
\sidenoteseq	\setsidenotsep
\startstanzahook	Optional argument of \stanza
\symplinenum	\Xsymplinenum

**Appendix A.9.5 Commands the names of which have been changed**

In order to help the migration from eledmac to reledmac, you may load reledmac with eledmac-compat option. However, it is advised not to, and to change the command names themselves instead. In many cases, you use only a few of them, except the \footparagraph command.

<i>Old command</i>	<i>New command</i>
\footparagraph	\Xarrangement
\footnormal	\Xarrangement
\foottwocol	\Xarrangement
\footthreecol	\Xarrangement
\footparagraphX	\arrangementX
\footnormalX	\arrangementX
\foottwocolX	\arrangementX
\footthreecolX	\arrangementX
\afterlemmaseparator	\Xafterlemmaseparator
\afternote	\Xafternote and \afternoteX
\afternumberinfootnote	\Xafternumber
\afterXrule	\Xafterrule
\afterXsymplinenum	\Xaftersymplinenum
\beforelemmaseparator	\Xbeforelemmaseparator
\beforenumberinfootnote	\Xbeforenumber
\beforeXnotes	\Xbeforenotes
\beforeXsymplinenum	\Xbeforesymplinenum



<i>Old command</i>	<i>New command</i>
<code>\bhookXnote</code>	<code>\Xbhookendnote</code>
<code>\bhookXnote</code>	<code>\Xbhooknote</code>
<code>\boxendlinenum</code>	<code>\Xboxendlinenum</code>
<code>\boxlinenum</code>	<code>\Xboxlinenum</code>
<code>\boxlinenumalign</code>	<code>\Xboxlinenumalign</code>
<code>\boxstartlinenum</code>	<code>\Xboxstartlinenum</code>
<code>\boxsymlinenum</code>	<code>\Xboxsymlinenum</code>
<code>\boxXendlinenum</code>	<code>\Xendboxlinenum</code>
<code>\boxXendlinenumalign</code>	<code>\Xendboxlinenumalign</code>
<code>\boxXendstartlinenum</code>	<code>\boxXendstartlinenum</code>
<code>\letboxXendendlinenum</code>	<code>\Xendletboxendlinenum</code>
<code>\hsizetwocol</code>	<code>\Xhsizetwocol</code>
<code>\hsizethreecol</code>	<code>\Xhsizethreecol</code>
<code>\inplaceoflemmaseparator</code>	<code>\Xinplaceoflemmaseparator</code>
<code>\inplaceofnumber</code>	<code>\Xinplaceofnumber</code>
<code>\lemmaseparator</code>	<code>\Xlemmaseparator</code>
<code>\maxhXnotes</code>	<code>\Xmaxhnotes</code>
<code>\morethantwolines</code>	<code>\Xmorethantwolines</code>
<code>\nonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\notesXwidthliketwocolumns</code>	<code>\noteswidthliketwocolumnsX</code>
<code>\noXlemmaseparator</code>	<code>\Xnolemmaseparator</code>
<code>\numberonlyfirstinline</code>	<code>\Xnumberonlyfirstinline</code>
<code>\numberonlyfirstintwolines</code>	<code>\Xnumberonlyfirstintwolines</code>
<code>\nonbreakableafternumber</code>	<code>\Xnonbreakableafternumber</code>
<code>\onlyXpstart</code>	<code>\Xonlypstart</code>
<code>\parafootsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\pstartinfootnote</code>	<code>\Xpstart</code>
<code>\pstartinfootnoteeverytime</code>	<code>\Xpstarteverytime</code>
<code>\symlinenum</code>	<code>\Xsymlinenum</code>
<code>\twolines</code>	<code>\Xtwolines</code>
<code>\twolinesbutnotmore</code>	<code>\Xtwolinesbutnotmore</code>
<code>\twolinesonlyinsamepage</code>	<code>\Xtwolinesonlyinsamepage</code>
<code>\txtbeforeXnotes</code>	<code>\Xtxtbeforenotes</code>
<code>\XendXafterlemmaseparator</code>	<code>\Xendafterlemmaseparator</code>
<code>\XendXbeforelemmaseparator</code>	<code>\Xendbeforelemmaseparator</code>
<code>\XendXinplaceoflemmaseparator</code>	<code>\Xendinplaceoflemmaseparator</code>
<code>\XendXlemmaseparator</code>	<code>\Xendlemmaseparator</code>
<code>\XendXmorethantwolines</code>	<code>\Xendmorethantwolines</code>
<code>\XendXtwolines</code>	<code>\Xendtwolines</code>
<code>\Xnonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\lineref</code>	<code>\edlineref</code>



### Appendix A.9.6 Endnotes

With *reledmac*, there is now one auxiliary file for every endnotes set (*.Aend*, *.Bend*, *.Cend* etc.). If you have overridden `\doendnotes` (which you would not have done) you must adapt your code.

### Appendix A.9.7 Z Series

The ‘Z’ series of notes has been removed. Only five series are provided now by default: A, B, C, D, E.

### Appendix A.9.8 Internal commands

Users who have overridden internal commands, which is wrong, must adapt according to the following. Or better, they should not override any of such commands and use *reledmac* options instead.

- If you have modified `\Xfootfmt`, note that the fourth argument is now mandatory.
- `\unvxh` has been replaced with `\Xunvxh` and `\unvxhX` with two mandatory arguments.

## Appendix A.10 Migration to *reledmac* 2.1.0

*Reledmac* 2.1.0 fix some bugs when using `\Xbhooknote` and `\bhooknoteX` not in order to execute code at the beginning of each notes, but to insert content of at the beginning of each notes.

People who use these commands to do it, which is not the original idea, must change the following:

1. Horizontal space is no longer automatically added after the content of the `\Xbhooknote/\bhooknoteX` argument. You must include it manually. So instead of `\Xbhooknote{content}`, use `\Xbhooknote{content }.`
2. Indent is no longer automatically added before the content of the `\Xbhooknote/\bhooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xbhooknote/\bhooknoteX`.

## Appendix A.11 Migration to *reledmac* 2.1.3

*Reledmac* 2.1.3 fix an historical bug, (style in *ledmac* 0.7!) which doubled the space before the rules of paragraphed familiar footnotes. Consequently, if you use paragraphed familiar footnotes, you should maybe adapt it, playing with `\beforenotesX`.

## Appendix A.12 Migration to *reledmac* 2.3.0

Before *reledmac* 2.3.0, for typesetting verse, any empty line was considered a paragraph inside verses. Counting empty lines this created breaking verse, hanging verses, and also added spurious vertical spaces. Version 2.3.0 disables paragraph in stanza. If you want vertical space, use optional argument of `\stanza` or `\endverse`.



**Appendix A.13 Migration to reledmac 2.4.0**

It is not mandatory, but strongly recommended, to change any `\renewcommand` `marg-{\dots}` to the use of `\Xlimerangeseparator` or `/` and `\Xendlimerangeseparator` (6.2.2 p. 30).



## 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 `edstanza.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> .....	40
<code>\@EDROWFILL@</code> .....	1
<code>\@adv</code> .....	1
<code>\@advancestanzanumber</code> .....	1
<code>\@apprefprefixmore</code> .....	1
<code>\@apprefprefixsingle</code> .....	1
<code>\@doclearpage</code> .....	1
<code>\@doreinfeetX</code> .....	1
<code>\@edindex@hyperref</code> .....	1
<code>\@edrowfill@</code> .....	1
<code>\@edtext@level</code> .....	1



\@emptytoks	1
\@fnpos	1
\@footnotemark	1
\@footnotetext	1
\@getfirstseries	1
\@gobblefive	1
\@gobblefour	1
\@gobblethree	1
\@h	1
\@hangingsymbol	1
\@iiiminipage	1
\@insertstanzaumber	1
\@k	1
\@l@dttempcnta	1
\@l@dttempcntb	1
\@lab	1
\@led@testifnofoot	1
\@lemma	1
\@line@num	1
\@lock	1
\@lopL	1
\@lopR	1
\@makechapterhead	1
\@makeschapterhead	1
\@mem@extranofeet	1
\@mem@old@ssect	1
\@mpfnpos	1
\@nl	1
\@nl@reg	1
\@opXfeet	1
\@pend	1
\@pendR	1
\@ref	1
\@ref@reg	1
\@sect	1
\@series	1
\@set	1
\@sidenotesep	1
\@ssect	1
\@startstanza	1
\@stopstanza	1
\@sw	1
\@tag	1
\@wredindex	1
\@xloop	1
\@xympar	1
CLASSbook	276
CLASSmemoir	168, 215, 216, 240, 276, 294, 348, 352
CLASSscrbook	352
COMMAND\*footnote	57



COMMAND\...\@footnotemark...	170
COMMAND\...d@ta	126
COMMAND\<hook	
@<series	205
COMMAND\<hookname	
<pseudoserious	207
COMMAND\<type	
footfmt	159
COMMAND\@@line	153
COMMAND\@MM	141, 349
COMMAND\@Rlineflag	241, 349
COMMAND\@add@	265
COMMAND\@adv	90
COMMAND\@apprefprefixmore	224, 225
COMMAND\@apprefprefixsingle	224, 225
COMMAND\@bsphack	217
COMMAND\@doclearpage	215, 216, 342, 352
COMMAND\@doreinfeetX	352
COMMAND\@dprintingcolumns	349
COMMAND\@edindex@hyperref	242, 243
COMMAND\@edtext@	106
COMMAND\@esphack	217
COMMAND\@fnpos	185
COMMAND\@footnotemark	168, 169, 342, 352
COMMAND\@footnotetext	168, 169, 342
COMMAND\@gobble	105
COMMAND\@gobblefive	202, 350
COMMAND\@gobblefour	348
COMMAND\@gobblethree	341
COMMAND\@h	155
COMMAND\@hangingsymbol	244
COMMAND\@iiiminipage	232, 234, 341, 352
COMMAND\@iiiminipage	232
COMMAND\@l	347
COMMAND\@l@tempcnta	128, 130, 136
COMMAND\@l@tempcntb	130
COMMAND\@l@reg	347
COMMAND\@lab	87, 217, 219, 220, 222, 341
COMMAND\@ldunboxmpfoot	234
COMMAND\@led@extranofeet	295
COMMAND\@ledinnote@command	238, 239
COMMAND\@lemma	109, 111
COMMAND\@lock	82, 244
COMMAND\@lopL	342
COMMAND\@lopR	342
COMMAND\@makecol	212, 215, 352
COMMAND\@mpfnpos	185
COMMAND\@nl	87–90, 92, 99, 220, 341, 342
COMMAND\@nl@reg	88, 292, 342, 347
COMMAND\@opXfeet	342



COMMAND\@opfeetX	352
COMMAND\@opxtrafeeti	352
COMMAND\@page	89, 220
COMMAND\@pend	342
COMMAND\@pendR	342
COMMAND\@ref	87, 96, 97, 100, 104
COMMAND\@ref@reg	96, 342
COMMAND\@reinserts	212, 213, 215, 352
COMMAND\@secondoftwo	57
COMMAND\@sect	279
COMMAND\@series	203
COMMAND\@set	91
COMMAND\@sidenotesep	231
COMMAND\@sw	97, 112, 115, 116
COMMAND\@tag	106, 107, 110
COMMAND\@tempcnta	68
COMMAND\@tempcntb	68
COMMAND\@toksa	74
COMMAND\@toksb	74
COMMAND\@xloop	137
COMMAND\@xympar	226, 352
COMMAND\Aendnote	14, 22
COMMAND\Afootfmt	140
COMMAND\Afootgroup	140
COMMAND\Afootnote	7, 14, 21, 22, 24, 108, 148, 168, 186, 198, 351
COMMAND\Afootstart	140
COMMAND\AtEveryPend	16, 121, 348, 350, 351
COMMAND\AtEveryPstart	16, 348, 350, 351
COMMAND\Bendnote	14, 21
COMMAND\Bfootnote	7, 14, 168, 186, 198
COMMAND\Centering	36
COMMAND\Cfootnote	168
COMMAND\Columns	69, 145
COMMAND\Dfootnote	168
COMMAND\Efootnote	168
COMMAND\NR	279
COMMAND\Pages	69, 213, 214
COMMAND\ProcessOptionsX	61
COMMAND\RaggedLeft	36
COMMAND\RaggedRight	36
COMMAND\Stanza	347
COMMAND\Waklam	266
COMMAND\X@doreinfeet	213, 352
COMMAND\XXXXXXfmt	291
COMMAND\XXXXXXfint	291
COMMAND\Xafterlemmaseparator	34, 295
COMMAND\Xafternote	37, 294, 295
COMMAND\Xafternumber	32, 295
COMMAND\Xafterrule	38, 186, 295, 347, 350
COMMAND\Xaftersymlinenum	32, 295



COMMAND\Xarrangement	29, 57, 141, 142, 205, 295
COMMAND\Xarrangement@footparagraph	146
COMMAND\Xarrangement@normal	142
COMMAND\Xarrangement@paragraph	146
COMMAND\Xbeforelemmaseparator	34, 295
COMMAND\Xbeforenotes	37, 186, 295, 347, 350
COMMAND\Xbeforenumber	32, 295
COMMAND\Xbeforesymlinenum	32, 295
COMMAND\Xbhookendnote	296
COMMAND\Xbhooknote	36, 296, 297, 352, 353
COMMAND\Xboxendlinenum	33, 296, 351
COMMAND\Xboxlinenum	33, 296
COMMAND\Xboxlinenumalign	33, 296, 351
COMMAND\Xboxstartlinenum	33, 296, 351
COMMAND\Xboxsymlinenum	33, 296
COMMAND\Xcolalign	36, 350
COMMAND\Xdo@feet	212, 342, 352
COMMAND\Xend	202
COMMAND\XendXafterlemmaseparator	296
COMMAND\XendXbeforelemmaseparator	296
COMMAND\XendXinplaceoflemmaseparator	296
COMMAND\XendXlemmaseparator	296
COMMAND\XendXmorethantwolines	296
COMMAND\XendXtwolines	296
COMMAND\Xendafterenumber	32, 353
COMMAND\Xendafterlemmaseparator	34, 296
COMMAND\Xendafternote	39
COMMAND\Xendafternumber	34
COMMAND\Xendaftersymlinenum	32, 34, 353
COMMAND\Xendahookinplaceofnumber	34, 353
COMMAND\Xendahooklinenumber	34, 353
COMMAND\Xendbeforelemmaseparator	34, 296
COMMAND\Xendbeforelinenumber	34
COMMAND\Xendbeforenumber	32, 353
COMMAND\Xendbeforesymlinenum	32, 34, 353
COMMAND\Xendbhookinplaceofnumber	34, 353
COMMAND\Xendbhooklinenumber	34, 353
COMMAND\Xendbhooknote	36
COMMAND\Xendboxendlinenum	33, 351
COMMAND\Xendboxlinenum	33, 296, 349
COMMAND\Xendboxlinenumalign	33, 296, 351
COMMAND\Xendboxstartlinenum	33, 351
COMMAND\Xendboxsymlinenum	33, 353
COMMAND\Xendhangindent	36, 353
COMMAND\Xendinplaceoflemmaseparator	35, 296
COMMAND\Xendinplaceofnumber	33, 352
COMMAND\Xendinsertsep@	195
COMMAND\Xendlemmadisablefontselection	35
COMMAND\Xendlemmaseparator	23, 34, 296
COMMAND\Xendletboxendlinenum	296



COMMAND\Xendlinangeseparator	30, 298, 353
COMMAND\Xendmorethantwolines	22, 31, 46, 296, 350
COMMAND\Xendmorethantwolinesappref	47
COMMAND\Xendnonumber	31, 352
COMMAND\Xendnote	189, 201, 202, 350
COMMAND\Xendnotefontsize	35, 295
COMMAND\Xendnotenumfont	34, 35, 295
COMMAND\Xendnumberonlyfirstinline	30, 353
COMMAND\Xendnumberonlyfirstintwolines	30, 353
COMMAND\Xendparagraph	39, 347
COMMAND\Xendsep	39
COMMAND\Xendsymlinenum	30, 353
COMMAND\Xendtwolines	22, 31, 46, 296, 350
COMMAND\Xendtwolinesappref	47
COMMAND\Xendtwolinesbutnotmore	31, 46, 350
COMMAND\Xendtwolinesbutnotmoreappref	47
COMMAND\Xendtwolinesonlyinsamepage	31, 46, 350
COMMAND\Xendtwolinesonlyinsamepageappref	47
COMMAND\Xfootfmt	297
COMMAND\Xfootgroup	145
COMMAND\Xfootins	144
COMMAND\Xfootnote	45, 106, 291, 344, 348–350
COMMAND\Xfootstarts	145
COMMAND\Xhangindent	36, 353
COMMAND\Xsizethreecol	37, 296
COMMAND\Xsizetwocol	37, 206, 296
COMMAND\Xinplaceoflemmaseparator	34, 296
COMMAND\Xinplaceofnumber	33, 296, 349, 351
COMMAND\Xinsertparafootsep	151, 152
COMMAND\Xledsetnormalparstuff	293, 294, 350
COMMAND\Xlemmadisablefontselection	35
COMMAND\Xlemmaseparator	34, 208, 210, 212, 296
COMMAND\Xlinangeseparator	30, 298, 353
COMMAND\Xmaxhnotes	38, 57, 295, 296, 347, 349
COMMAND\Xmorethantwolines	22, 31, 46, 224, 296, 349
COMMAND\Xmorethantwolinesappref	46, 224
COMMAND\Xnoindent	353
COMMAND\Xnolemmaseparator	34, 212, 296
COMMAND\Xnonbreakableafternumber	32, 296, 345
COMMAND\Xnonumber	31, 296
COMMAND\Xnonumberinfootnote	296
COMMAND\Xnotefontsize	35, 295
COMMAND\Xnotefontsize@(<s>)	151, 154, 155
COMMAND\Xnotenumfont	35, 295
COMMAND\Xnoteswidthliketwocolumns	38, 348
COMMAND\Xnumberonlyfirstinline	30, 84, 207, 208, 210, 296, 344, 349
COMMAND\Xnumberonlyfirstintwolines	30, 296, 344
COMMAND\Xonlypstart	32, 296, 344, 349
COMMAND\Xparafootsep	37, 85, 295, 296
COMMAND\Xparafootsep@series	151



COMMAND\Xparindent	35, 350, 353
COMMAND\Xpstart	31, 32, 293, 296, 344, 349
COMMAND\Xpstarteverytime	32, 293, 296, 349
COMMAND\Xragged	37
COMMAND\Xstanza	32, 42
COMMAND\Xstanzaseparator	32
COMMAND\Xsymlinenum	30, 37, 295, 296, 351
COMMAND\Xtwolines	22, 31, 46, 165, 207, 224, 296, 349
COMMAND\Xtwolinesappref	46, 207, 224
COMMAND\Xtwolinesbutnotmore	31, 46, 296, 350
COMMAND\Xtwolinesbutnotmoreappref	46, 207
COMMAND\Xtwolinesonlyinsamepage	31, 46, 296, 350
COMMAND\Xtwolinesonlyinsamepageappref	46
COMMAND\Xtxtbeforenotes	37, 296
COMMAND\Xunvxh	148, 297
COMMAND\&	295
COMMAND\absline@num	81, 127
COMMAND\accent	105
COMMAND\actionlines@list	82, 129
COMMAND\actions@list	82
COMMAND\add@inserts	82, 135
COMMAND\add@inserts@next	135, 136
COMMAND\add@penalties	127, 136
COMMAND\addcontentsline	273
COMMAND\addfootins	291, 294
COMMAND\addfootinsX	291, 294
COMMAND\advancelabel@refs	219
COMMAND\advanceline	20, 84, 90, 101, 352
COMMAND\advancepageno	212
COMMAND\affixlin@num	231
COMMAND\affixline@num	130, 132, 134, 342
COMMAND\affixpstart@num	134
COMMAND\afterXrule	295
COMMAND\afterXsymlinenum	295
COMMAND\afterenumber	32
COMMAND\aftergroup	104, 108
COMMAND\afterlemmaseparator	295
COMMAND\afternote	295
COMMAND\afternoteX	37, 294, 295
COMMAND\afternumberinfootnote	295
COMMAND\afterruleX	38, 347, 350
COMMAND\applabel	46, 220, 221, 224, 350
COMMAND\appref	46, 224, 225
COMMAND\apprefprefixmore	295
COMMAND\apprefprefixsingle	295
COMMAND\apprefwithpage	46, 47, 224, 225
COMMAND\arrangementX	29, 57, 170, 205, 295
COMMAND\arrangementX@normal	175
COMMAND\at@every@pend	121
COMMAND\autopar	16, 118, 122, 123, 183, 343, 345, 346, 350



COMMAND\ballast	56
COMMAND\ballast@count	127, 136
COMMAND\baselineskip	30, 147, 151
COMMAND\beforeXnotes	295
COMMAND\beforeXsymlinenum	295
COMMAND\beforeelectedchapter	9, 54, 274, 275
COMMAND\beforelemmaseparator	295
COMMAND\beforenotesX	37, 297, 346, 347, 350
COMMAND\beforenumberinfootnote	295
COMMAND\begin	251
COMMAND\beginnumbering	14–17, 69, 70, 72, 81, 85, 99, 122, 189, 344, 347, 351, 352
COMMAND\bf	344
COMMAND\bfseries	35, 344
COMMAND\bhookXnote	296
COMMAND\bhooknoteX	36, 297, 352, 353
COMMAND\body	245
COMMAND\bodyfootmarkA	27
COMMAND\boxXendlinenum	296
COMMAND\boxXendlinenumalign	296
COMMAND\boxXendstartlinenum	296
COMMAND\boxendlinenum	296
COMMAND\boxlinefootnote	162
COMMAND\boxlinenum	296
COMMAND\boxlinenumalign	296
COMMAND\boxstartlinenum	296
COMMAND\boxsymlinenum	296
COMMAND\break	30, 149
COMMAND\brokenpenalty	136
COMMAND\centering	36
COMMAND\ch@ck@l@ck	343
COMMAND\ch@cksub@l@ck	132, 343
COMMAND\chapter	53, 276, 347, 350, 352
COMMAND\chaptermark	274
COMMAND\check@pb@in@verse	287
COMMAND\colalignX	36, 350
COMMAND\collect@body	252
COMMAND\colorbox	57
COMMAND\columns	38
COMMAND\columnwidth	147, 348
COMMAND\command names	206, 207
COMMAND\copyright	105
COMMAND\correct@Xfootins@box	349
COMMAND\correct@footinsX@box	349
COMMAND\count	154
COMMAND\critex	343
COMMAND\critext	111, 290, 291, 294
COMMAND\csname	61, 114
COMMAND\ctab	267, 272
COMMAND\ctabtext	272
COMMAND\dcoll	261



COMMAND\def	59
COMMAND\detokenize	114
COMMAND\dimen	154
COMMAND\discretionary	148
COMMAND\displaywidowpenalty	136
COMMAND\do@actions	127–129, 343
COMMAND\do@actions@fixedcode	342
COMMAND\do@actions@next	128, 129
COMMAND\do@ballast	127, 136
COMMAND\do@feetX	352
COMMAND\do@insidelinehook	345
COMMAND\do@line	82, 104, 120, 123, 126, 135, 136, 244, 343, 345, 347
COMMAND\do@linehook	343
COMMAND\do@lockoff	83
COMMAND\do@lockon	83
COMMAND\dodoreintrafeet	341
COMMAND\doendnotes	23, 195, 297, 350
COMMAND\doendnotesbysection	23, 195, 202, 351
COMMAND\doinsidelinehook	21, 348
COMMAND\dolinehook	20, 348
COMMAND\doreintrafeeti	352
COMMAND\doreintrafeetii	352
COMMAND\doxtrafeet	212, 341
COMMAND\doxtrafeeti	352
COMMAND\doxtrafeetii	352
COMMAND\dummy@ref	104
COMMAND\edaftertab	52, 267, 268
COMMAND\edatleft	52, 265
COMMAND\edatright	52, 265
COMMAND\edbeforetab	52, 267
COMMAND\edfilldimen	265
COMMAND\edfont@info	110
COMMAND\edindex	49, 237, 241–243, 255, 345, 348, 349, 352, 353
COMMAND\edindexlab	49
COMMAND\edlabel	44–46, 105, 217, 218, 220, 221, 226, 237, 255, 341, 344–346, 349
COMMAND\edlineref	44, 217, 296, 349, 351
COMMAND\edmakelabel	45, 46, 226
COMMAND\edpageref	44, 217, 221, 226
COMMAND\edrowfill	266
COMMAND\edtabcolsep	260
COMMAND\edtext	6, 21, 22, 24, 26, 27, 39, 44–46, 50, 57, 82, 96, 97, 100, 103–111, 113–117, 220, 221, 223, 255, 257, 276, 290, 291, 294, 342, 343, 345, 347–351
COMMAND\edtext@level	351
COMMAND\edvertdots	53, 265
COMMAND\edvertline	52, 53, 265
COMMAND\elechapter	54
COMMAND\eled@sectioning@out	280
COMMAND\eledchapter	53, 294, 348, 352
COMMAND\eledchapter*	53
COMMAND\eledmac@error	341



COMMAND\eledsection	6, 14, 53, 105, 125, 274, 294, 349
COMMAND\eledsection*	53
COMMAND\eledsubsection	53, 294
COMMAND\eledsubsection*	53
COMMAND\eledsubsubsection	53, 294
COMMAND\eledsubsubsection*	53
COMMAND\eledxxx	9, 54, 275, 280, 347
COMMAND\eledxxxx	274
COMMAND\else	236, 274
COMMAND\empty	68, 131, 217
COMMAND\end	251
COMMAND\end@lemmas	104
COMMAND\endashchar	39, 159
COMMAND\endgraf	120, 150, 183
COMMAND\endlock	19, 83, 102, 249
COMMAND\endminipage	232, 234, 341, 352
COMMAND\endnotes	350
COMMAND\endnumbering	14, 15, 17, 69–72, 343, 351
COMMAND\endprint	189, 191, 202, 293
COMMAND\endstanzaextra	295
COMMAND\endsub	19, 83, 101
COMMAND\endverse	297
COMMAND\everypar	122
COMMAND\extensionchars	55, 69
COMMAND\f@x@l@cks	343
COMMAND>falseverse	294, 345, 347
COMMAND\fi	274
COMMAND\firstlinenum	18, 130, 343
COMMAND\firstsublinenum	18, 343
COMMAND\fix@page	88, 89, 342
COMMAND\flag@end	100, 109, 347
COMMAND\flag@start	100, 109, 347, 348
COMMAND\flagstanza	43
COMMAND\floatingpenalty	141, 349
COMMAND\flush@notes	137
COMMAND\fnpos	185, 346
COMMAND\footfmt	140, 143
COMMAND\footfmt...	171
COMMAND\footfootmarkA	28
COMMAND\footfudgefactor	149
COMMAND\footfudgefiddle	57, 147, 341
COMMAND\footgroup	140
COMMAND\footins	144
COMMAND\footnormal	206, 295, 342
COMMAND\footnormalX	295
COMMAND\footnote	27, 56, 168, 169, 292, 342
COMMAND\footnote@lang	159
COMMAND\footnoteA	14, 27
COMMAND\footnoteB	14
COMMAND\footnoteC	21



COMMAND\footnoteE	27
COMMAND\footnoteX	7, 200
COMMAND\footnoteXmk	211
COMMAND\footnotelang@lua	139
COMMAND\footnotelang@poly	139
COMMAND\footnoteoption@	138, 353
COMMAND\footnoterule	154
COMMAND\footnotesize	35
COMMAND\footparagraph	147, 206, 295, 347
COMMAND\footparagraphX	180, 295, 347
COMMAND\footplitskips	343, 349
COMMAND\footstart	140, 144, 154
COMMAND\footstrut	150
COMMAND\footthreecol	295
COMMAND\footthreecolX	295, 350
COMMAND\foottwocol	295
COMMAND\foottwocolX	295, 350
COMMAND\fullstop	39
COMMAND\get@edindex@hyperref	242
COMMAND\get@edindex@ledinnote@command	238
COMMAND\get@index@command	346
COMMAND\get@linelistfile	343
COMMAND\getline@num	127, 128
COMMAND\gl@p	74
COMMAND\global	87
COMMAND\globaldefs	87
COMMAND\hangindentX	36, 350, 353
COMMAND\hangingsymbol	295, 343
COMMAND\hbox	148
COMMAND\hfill	346
COMMAND\hidenumbering	20, 95, 350
COMMAND\hline	50
COMMAND\hrulefill	266
COMMAND\hsizex	30, 144, 147, 149, 154, 157, 184, 342, 348
COMMAND\hsizethreecol	296
COMMAND\hsizethreecolX	37
COMMAND\hsizetwocol	296
COMMAND\hsizetwocolX	37
COMMAND\hyperlinkR	241
COMMAND\hyperlinkformat	241
COMMAND\hyperlinkformatR	241
COMMAND\if@RTL	63
COMMAND\if@edtext@	348, 351
COMMAND\if@eled@sectioning	275
COMMAND\if@noneed@Footnote	100
COMMAND\ifbypage@	74
COMMAND\ifbypage@R	75
COMMAND\ifbypstart@	74
COMMAND\ifbypstart@R	75
COMMAND\iffirst@linenum@out@	98, 99



COMMAND\ifinserthangingsymbol	244
COMMAND\ifinstanza	244
COMMAND\ifstwofollowinglines	165, 166
COMMAND\ifl@d@Xmorethantwolines	163, 350
COMMAND\ifl@d@Xtwolines	163
COMMAND\ifl@d@dash	163
COMMAND\ifl@d@elin	163
COMMAND\ifl@d@esl	163
COMMAND\ifl@d@pnum	163
COMMAND\ifl@d@ssub	163
COMMAND\ifl@dend@X	201
COMMAND\ifl@dmemoir	341
COMMAND\ifl@dpaging	348
COMMAND\ifl@dpairing	69, 343
COMMAND\ifl@dprintingpages	349
COMMAND\ifl@dskipnumber	130
COMMAND\ifl@dstartendok	266
COMMAND\ifl@imakeidx	62
COMMAND\ifledRcol	69, 344
COMMAND\ifledRcol@	69, 347
COMMAND\iflemmacommand@	349
COMMAND\ifnoledgroup@	236
COMMAND\ifnoteschanged@	84
COMMAND\ifnumberedpar@	118
COMMAND\ifnumbering	70, 72
COMMAND\ifnumberingR	69, 344
COMMAND\ifnumberline	109, 130
COMMAND\ifpst@rted	343
COMMAND\ifpst@rtedL	70
COMMAND\ifseriesbefore	204
COMMAND\ifsublines@	81, 93
COMMAND\iftrue	351
COMMAND\ifvmode	219
COMMAND\ifxxx	274
COMMAND\ignorespaces	108
COMMAND\imki@wrindexentry	62
COMMAND\immediate	98, 99, 188
COMMAND\indent	16, 122, 297
COMMAND\indtl@wrindexentry	62
COMMAND\initnumbering@quote	272, 352
COMMAND\initnumbering@reg	343
COMMAND\initnumbering@sectcmd	352
COMMAND\inplaceoflemmaseparator	296
COMMAND\inplaceofnumber	296
COMMAND\insert	135, 140, 143, 171
COMMAND\insert@count	96, 100, 107
COMMAND\insert@countR	107
COMMAND\inserthangingsymbol	346
COMMAND\insertlines@list	82, 96
COMMAND\insertparafootsepX	183



COMMAND\inserts@list	104, 118, 135, 148
COMMAND\interAfootnotelinepenalty	342
COMMAND\interfootnotelinepenalty	342
COMMAND\interlinepenalty	140
COMMAND\interparanoteglue	294
COMMAND\justifying	36
COMMAND\l@advance@parledegroupp@beforenormalnotes	352
COMMAND\l@d@@wrindexhyp	348
COMMAND\l@d@add	112
COMMAND\l@d@end	189, 201
COMMAND\l@d@nums	107, 109, 111, 112, 162, 163
COMMAND\l@d@section	189
COMMAND\l@d@set	92, 102
COMMAND\l@dampcount	257
COMMAND\l@dbfnote	169, 342
COMMAND\l@dcheckstartend	266
COMMAND\l@dchset@num	92
COMMAND\l@dcolcount	257, 258
COMMAND\l@dcollect@@body	251
COMMAND\l@dcollect@body	251
COMMAND\l@dcsnote	347
COMMAND\l@dcsnotetext	126, 229, 230
COMMAND\l@dcsnotetext@l	126, 229, 230
COMMAND\l@dcsnotetext@r	126, 229, 230
COMMAND\l@ddodoreintrafeet	213, 341
COMMAND\l@ddoxtrafeet	213, 341
COMMAND\l@demptyd@ta	343
COMMAND\l@dend@close	188
COMMAND\l@dend@open	188
COMMAND\l@dend@stuff	189
COMMAND\l@denvbody	251
COMMAND\l@dfeetbeginmini	342
COMMAND\l@dfeetendmini	342
COMMAND\l@dgetline@margin	343
COMMAND\l@dgetlock@disp	343
COMMAND\l@dgetref@num	222, 223
COMMAND\l@dgetsidenote@margin	226, 343
COMMAND\l@dgobbeloptarg	348
COMMAND\l@dgobblearg	348
COMMAND\l@dgobbleoptarg	256
COMMAND\l@dlabel@parse	222, 223
COMMAND\l@dld@ta	130, 132
COMMAND\l@dlp@rbox	231
COMMAND\l@dlsn@te	343
COMMAND\l@dlsnote	347
COMMAND\l@dmake@labels	219
COMMAND\l@dnumpsstartsL	70, 343
COMMAND\l@dp@rsefootspec	163
COMMAND\l@dparsefootspec	163
COMMAND\l@dpush@begins	251



COMMAND\l@drd@ta	130, 132
COMMAND\l@dref@undefined	222
COMMAND\l@drsn@te	343
COMMAND\l@drsnote	347
COMMAND\l@dtabaddcols	265
COMMAND\l@dtabnoexpands	341
COMMAND\l@dumboxmpfoot	352
COMMAND\l@dunboxmpfoot	343
COMMAND\l@dzeropenalties	343, 348
COMMAND\l@pb	286
COMMAND\l@prev@nopb	285
COMMAND\l@prev@pb	285
COMMAND\l@reg	292
COMMAND\label	17, 46, 49, 217, 218, 223
COMMAND\label@refs	217
COMMAND\labelstarttrue	17, 344
COMMAND\labelref@list	217, 219, 220
COMMAND\language	148
COMMAND\last@page@num	342
COMMAND\lastbox	122
COMMAND\lastskip	101
COMMAND\leavevmode	16, 122
COMMAND\led@check@nopb	286
COMMAND\led@check@pb	286
COMMAND\led@nopb	285–287
COMMAND\led@nopbnum	285, 286
COMMAND\led@pb	285–287
COMMAND\led@pb@macro	286
COMMAND\led@pbnum	285, 286
COMMAND\ledRflag	241
COMMAND\ledchapter	294, 345
COMMAND\ledfootinsdim	295
COMMAND\ledinnernote	47, 228, 347
COMMAND\ledinnote	239, 293, 351
COMMAND\ledinnotemark	45, 293, 350
COMMAND\ledleftnote	47, 228
COMMAND\ledlinenum	80, 343
COMMAND\ledllfill	126
COMMAND\ledlsnotesep	47
COMMAND\ledlsnotewidth	47
COMMAND\lednopb	55, 285
COMMAND\lednopbinverse	287
COMMAND\lednopbinversetrue	42, 55
COMMAND\lednopbnum	285
COMMAND\ledouternote	47, 228, 347
COMMAND\ledpb	55, 285, 294
COMMAND\ledpbnum	285
COMMAND\ledpbsetting	55, 286, 352
COMMAND\ledrightnote	47, 228
COMMAND\ledrsnotesep	47



COMMAND\ledrsnotewidth	47
COMMAND\ledsection	294
COMMAND\ledsectnomark	274
COMMAND\ledsectnotoc	273
COMMAND\ledsetnormalparstuff	293, 294, 350
COMMAND\ledsetnormalparstuff@common	183
COMMAND\ledsetnormalparstuffX	293, 294, 350
COMMAND\ledsidenote	47, 228–230
COMMAND\ledsubsection	294
COMMAND\ledsubsubsection	294
COMMAND\ledxxx	347
COMMAND\left	52
COMMAND\leftctab	267
COMMAND\leftheadline	80
COMMAND\leftlinenum	19, 80, 341, 343
COMMAND\leftltab	267
COMMAND\leftnoteupfalse	47
COMMAND\leftpstartnum	134
COMMAND\leftrtab	267
COMMAND\leftsidenote	230
COMMAND\leftskip	144, 147, 149
COMMAND\lemma	2, 22, 24, 26, 27, 103, 106, 108, 110, 111, 113, 290, 343, 344, 351, 352
COMMAND\lemmaseparator	296
COMMAND\let	25, 40, 249, 341
COMMAND\letboxXendendlinenum	296
COMMAND\line	153, 155
COMMAND\line@list	82, 97, 109, 110
COMMAND\line@list@stuff	70, 85, 99, 341, 343
COMMAND\line@list@version	87
COMMAND\line@margin	76, 132, 226
COMMAND\line@num	81, 83, 130, 341
COMMAND\line@set	111, 112
COMMAND\lineation	18, 75
COMMAND\linebreak	30
COMMAND\linenum	22, 24, 45, 46, 103, 111, 221, 223, 226, 290
COMMAND\linenum@out	98, 217, 219, 220
COMMAND\linenumberlist	18, 68, 131, 341
COMMAND\linenumberstyle	20, 79, 341
COMMAND\linenumincrement	18, 343
COMMAND\linenummargin	18, 19, 76, 226
COMMAND\linenumr@p	79, 341, 343
COMMAND\linenumrep	79, 343
COMMAND\linenumsep	19, 47, 80, 227
COMMAND\linerangesep@	211
COMMAND\lineref	217, 222, 226, 296, 349
COMMAND\list@clear	73
COMMAND\list@clearing@reg	343
COMMAND\list@create	73
COMMAND\lock@disp	78
COMMAND\lock@off	94



COMMAND\lock@on	93
COMMAND\lockdisp	19, 78
COMMAND\loop	137, 138, 245
COMMAND\ltab	267, 268, 272
COMMAND\ltabtext	272
COMMAND\m@mmf@prepare	169
COMMAND\makeatletter	126
COMMAND\makeboxofhboxes	149, 151
COMMAND\makeindex	48, 241
COMMAND\makelabel	226
COMMAND\managestanza@modulo	246
COMMAND\marginpar	47, 56, 226, 342
COMMAND\marginparwidth	47, 227
COMMAND\markboth	126
COMMAND\mathchardef	245
COMMAND\maxhXnotes	296
COMMAND\maxhnotesX	38, 57, 295, 346, 347, 349, 350
COMMAND\maxlinesinpar@list	85
COMMAND\measurebody	268
COMMAND\measuredbody	269
COMMAND\memorybreak	17
COMMAND\morenoexpands	57, 58, 104, 105
COMMAND\morethantwolines	296
COMMAND\mpfnpos	185, 346
COMMAND\mpnormalfootgroup	342
COMMAND\mpnormalvfootnote	342
COMMAND\multfootsep	28, 168
COMMAND\multiplefootnotemarker	168
COMMAND\musixtex	347
COMMAND\n@num	343, 350
COMMAND\n@num@ref	350
COMMAND\new@line	99, 342
COMMAND\newcommand	25, 59, 168, 219
COMMAND\newcommandx	25
COMMAND\newhookarg@specific	211
COMMAND\newhookcommand@series	206, 208, 350
COMMAND\newhookcommand@series@reload	208
COMMAND\newhookcommand@toggle@reload	207, 348
COMMAND\newhooktoggle@series	207, 350
COMMAND\newhooktoggle@specific	211
COMMAND\newif	350
COMMAND\newline	30
COMMAND\newlinechar	201
COMMAND\newseries	28, 291, 294, 295
COMMAND\newseries@	196, 205
COMMAND\newverse	42, 43, 294, 347
COMMAND\next	245
COMMAND\next@action	86
COMMAND\next@actionline	86
COMMAND\next@insert	136



COMMAND\nl@regR	88
COMMAND\no@expands	57, 110, 341
COMMAND\noXlemmaseparator	296
COMMAND\nobreak	162
COMMAND\nocritical	196
COMMAND\noeledsec	54, 294
COMMAND\noendnotes	294
COMMAND\noexpand	292
COMMAND\nofamiliar	209
COMMAND\noindent	16, 122, 353
COMMAND\noindentX	353
COMMAND\nomk@	211
COMMAND\nonbreakableafternumber	296
COMMAND\nonumberinfootnote	296
COMMAND\normal@footnotemarkX	171
COMMAND\normal@page@break	285
COMMAND\normal@pars	183
COMMAND\normalbfnoteX	343
COMMAND\normalbodyfootmarkX	171
COMMAND\normalfootfmt	40, 143, 150, 159, 189
COMMAND\normalfootfmtX	172
COMMAND\normalfootfootmarkX	172
COMMAND\normalfootgroup	145
COMMAND\normalfootgroupX	173
COMMAND\normalfootnoterule	141
COMMAND\normalfootstart	144, 147
COMMAND\normalfootstartX	172
COMMAND\normalvfootnote	143
COMMAND\normalvfootnoteX	171
COMMAND\notbool	274
COMMAND\notefontsetup	295
COMMAND\notefontsizeX	35, 295
COMMAND\notenumfont	295
COMMAND\notenumfontX	35, 295
COMMAND\notesXwidthliketwocolumns	296
COMMAND\noteswidthliketwocolumnsX	39, 296, 348, 350
COMMAND\num@lines	118, 136
COMMAND\numberlinefalse	18
COMMAND\numberlinetrue	18
COMMAND\numberonlyfirstinline	204, 296
COMMAND\numberonlyfirstintwolines	296
COMMAND\numberpstartfalse	16
COMMAND\numberpstarttrue	16, 32, 293, 343, 352
COMMAND\numberstanza	32
COMMAND\numberstanzafalse	42
COMMAND\numberstanzatrue	42
COMMAND\numlabfont	19, 39, 80
COMMAND\none@line	118
COMMAND\nonehalfspacing	353
COMMAND\nonlyXpstart	296



COMMAND\page@action	83, 92
COMMAND\page@start	83, 343
COMMAND\pagecontents	83
COMMAND\pagelinesep	48
COMMAND\pageno	212
COMMAND\pageparbreak	294
COMMAND\pageref	46, 221
COMMAND\par	23, 30, 122, 183
COMMAND\par@line	118, 136
COMMAND\para@footgroup	147
COMMAND\para@footgroupX	182
COMMAND\para@footsetup	147, 341
COMMAND\para@footsetupX	180, 341, 348
COMMAND\para@vfootnote	151
COMMAND\para@vfootnoteX	181
COMMAND\parafootfmt	149, 150
COMMAND\parafootfmtX	182
COMMAND\parafootftm	152
COMMAND\parafootftmX	183
COMMAND\parafootftmsep	295
COMMAND\parafootsep	296, 346, 351
COMMAND\parafootsepX	37, 85, 295, 296
COMMAND\parafootstart	147
COMMAND\parafootstartX	180
COMMAND\paravfootnote	148
COMMAND\parfillskip	150
COMMAND\parindent	353
COMMAND\parindentX	36, 353
COMMAND\parshape	56
COMMAND\parskip	122
COMMAND\pausenumbering	17, 72, 85, 87, 123, 346, 348
COMMAND\penalty	150
COMMAND\pend	2, 6, 15–17, 19, 20, 54, 101, 104, 106, 112, 118–123, 134, 135, 292, 346, 347
COMMAND\preXnotes	38, 187, 350
COMMAND\preXnotes@	144, 187, 344
COMMAND\prenotesX	38, 187, 350
COMMAND\prepare@preXnotes	186
COMMAND\prev@nopb	286
COMMAND\prev@pb	286
COMMAND\prevlineX	84
COMMAND\prevpageX@num	85
COMMAND\print@Xfootnoterule	350
COMMAND\print@Xnotes	213, 214
COMMAND\print@Xnotes@forpages	349
COMMAND\print@eledsection	125
COMMAND\print@footnoteXrule	350
COMMAND\print@leftmargin@eledsection	275
COMMAND\print@line	124
COMMAND\print@notesX@forpages	349
COMMAND\print@rightmargin@eledsection	275



COMMAND\printendlines	192, 225, 341, 343
COMMAND\printlinefootnote	159, 161, 349
COMMAND\printlinefootnotearea	161, 162, 349
COMMAND\printlinefootnotenumbers	159
COMMAND\printlines	143, 159, 162, 164, 192, 225, 341, 343, 350
COMMAND\printnpnum	23, 293
COMMAND\printpstart	159
COMMAND\protect	105, 292
COMMAND\providecommand	168, 341
COMMAND\pstart	2, 6, 15–17, 19, 20, 53, 54, 92, 101, 102, 106, 112, 118–122, 125, 135, 292, 343, 344, 346–348, 350–352
COMMAND\pstartinfootnote	296
COMMAND\pstartinfootnoteeverytime	296
COMMAND\pstartnum	134
COMMAND\pstartref	44, 217, 222, 346
COMMAND\pstarts	344
COMMAND\raggedX	37
COMMAND\raggedleft	36
COMMAND\raggedright	36
COMMAND\raw@text	118
COMMAND\rbracket	34, 39
COMMAND\read@linelist	85–87
COMMAND\ref	46, 49
COMMAND\relax	16, 92, 128, 135, 249, 256, 292
COMMAND\renewcommand	57, 295, 298
COMMAND\resetprevline@	84
COMMAND\resetprevpage@	85
COMMAND\resumenumbering	17, 69, 72, 85, 87, 123, 343, 347, 348
COMMAND\right	52
COMMAND\rightctab	267
COMMAND\rightlinenum	19, 80, 341, 343
COMMAND\rightltab	268
COMMAND\rightnoteupfalse	47
COMMAND\rightrtab	268
COMMAND\rightrightnote	229
COMMAND\rightrightskip	144, 147, 149, 150
COMMAND\rightrightstartnum	134
COMMAND\rigidbalance	153–155
COMMAND\robustify	30
COMMAND\rtab	267, 268, 272
COMMAND\rtabtext	269, 272
COMMAND\sameword	25–27, 112–114, 116, 349, 351, 353
COMMAND\sameword@inedtext	113
COMMAND\saweword	113
COMMAND\scriptsize	80
COMMAND\section	53, 343
COMMAND\section@num	69
COMMAND\sectionmark	274
COMMAND\select@lemmafont	39, 40, 138
COMMAND\series	196



COMMAND\series@	196
COMMAND\seriesatbegin	28, 203, 350
COMMAND\seriesatend	28, 204, 351
COMMAND\set@line	109
COMMAND\set@line@action	83, 92
COMMAND\setapprefprefixmore	46, 295
COMMAND\setapprefprefixsingle	46, 295
COMMAND\setcommand@series	206
COMMAND\sethangingsymbol	42, 244, 295, 353
COMMAND\sethanginsymbol	40
COMMAND\setistwofollowinglines	165
COMMAND\setl@dlprbox	231
COMMAND\setline	20, 84, 88, 91, 101, 105, 120, 352
COMMAND\setlinenum	20, 88, 92, 102, 341
COMMAND\setprintendlines	192, 193, 343
COMMAND\setprintlines	164, 165, 192, 343
COMMAND\setsidenotesepp	48
COMMAND\setsidenotsep	295
COMMAND\setstanzaindent	246
COMMAND\setstanzaindents	41, 245, 292
COMMAND\setstanzapenalties	245
COMMAND\setstanzavalues	245
COMMAND\settoggle@series	205, 344, 348
COMMAND\showlemma	105, 342
COMMAND\showwordrank	27, 114
COMMAND\sidenote@margin	342
COMMAND\sidenotemargin	47, 342, 347
COMMAND\sidenotesepp	295
COMMAND\sidepstartnumtrue	17
COMMAND\skip	144
COMMAND\skipnumbering	20, 95, 102, 343, 351
COMMAND\skipnumbering@reg	351
COMMAND\small	35
COMMAND\special	11
COMMAND\splitmaxdepth	141, 154
COMMAND\splitoff	153
COMMAND\splittopskip	140, 154, 155
COMMAND\stanza	19, 20, 42, 43, 248, 295, 297, 353
COMMAND\stanza@hang	248
COMMAND\stanza@line	248
COMMAND\stanzaindent	41, 246, 349
COMMAND\stanzaindent*	41
COMMAND\stanzaindentbase	245
COMMAND\stanzanumwrapper	43
COMMAND\startlock	19, 83, 102, 249
COMMAND\startstanzahook	295
COMMAND\startsub	19, 83, 101
COMMAND\strip@pt	147
COMMAND\strutbox	154
COMMAND\sub@action	83, 93



COMMAND\sub@lock	82
COMMAND\sub@off	90, 220
COMMAND\sub@on	90, 220
COMMAND\subline@num	81, 83
COMMAND\sublinenum@rep	341
COMMAND\sublinenumberstyle	20, 79, 341
COMMAND\sublinenumincrement	18
COMMAND\sublinenumr@p	79, 341, 343
COMMAND\sublinenumrep	79, 343
COMMAND\sublineref	44, 217, 222
COMMAND\subsectionmark	274
COMMAND\sw@inthisedtext	107
COMMAND\sw@list@inedtext	110, 117
COMMAND\symlinenum	296
COMMAND\symplinenum	295
COMMAND\sza@penalty	248
COMMAND>tag	349
COMMAND\text	290
COMMAND\textcolor	58
COMMAND\textheight	57
COMMAND\the	341
COMMAND\thefootnoteA	27
COMMAND\thefootnoteX	345
COMMAND\thelabidx	242, 243
COMMAND\thepage	88
COMMAND\thepstart	16
COMMAND\thepstartL	344
COMMAND\thepstartR	344
COMMAND\thestanza	42
COMMAND\this@line@list@version	98
COMMAND\threecolfootfmt	155
COMMAND\threecolfootfmtX	178
COMMAND\threecolfootgroup	154
COMMAND\threecolfootgroupX	179
COMMAND\threecolfootsetup	154
COMMAND\threecolfootsetupX	178
COMMAND\threecolvfootnote	154
COMMAND\threecolvfootnoteX	178
COMMAND\twocolfootfmtX	176
COMMAND\twocolfootgroupX	177
COMMAND\twocolfootsetupX	176
COMMAND\twocolvfootnoteX	176
COMMAND\twolines	205, 296
COMMAND\twolines@A	205
COMMAND\twolines@B	205
COMMAND\twolines@C	205
COMMAND\twolinesbutnotmore	296
COMMAND\twolinesonlyinsamepage	296
COMMAND\txbeforeXnotes	296
COMMAND\unhbox	148



COMMAND\unpenalty	149–151
COMMAND\unskip	150
COMMAND\unvxh	150, 297
COMMAND\unvxhX	297
COMMAND\upbracefill	266
COMMAND\usingcritext	291, 294
COMMAND\usingedtext	291, 294
COMMAND\VAfootnote	140
COMMAND\variant	24
COMMAND\ vbox	120, 122, 148, 149, 153, 185
COMMAND\ vfootnote	140, 144, 148, 154
COMMAND\ vl@dbfnote	169, 342
COMMAND\ vnumfootnoteX	343
COMMAND\ vsize	38, 57
COMMAND\ vsplit	136
COMMAND\ wklam	266
COMMAND\ wklamec	266
COMMAND\ wapunktel	266
COMMAND\ wastricht	266
COMMAND\ wrap@edcrossref	221, 348
COMMAND\ x...	45
COMMAND\ xdef	74, 249
COMMAND\ xleft@appenditem	74, 104
COMMAND\ xlineref	45
COMMAND\ xpageref	45
COMMAND\ xpstartref	45, 346
COMMAND\ xright@appenditem	74
COMMAND\ xsublineref	45
COMMAND\ xxref	45, 223, 226, 346, 349, 350
COMMAND\ zz@@@	341
ENVIRONMENTedarrayc	272
ENVIRONMENTedarrayl	272
ENVIRONMENTedarrayr	272
ENVIRONMENTedtabularc	272
ENVIRONMENTedtabularl	272
ENVIRONMENTedtabularr	272
ENVIRONMENTledgroup	235
ENVIRONMENTledgroupsized	235
PACKAGE(r)(e)ledmac	28
PACKAGEEledmac	10, 59, 83, 240, 293, 294, 349–351
PACKAGEEledpar	350, 351
PACKAGEEtoolbox	61
PACKAGEParallel	299
PACKAGEReledmac	297
PACKAGEamsgen	250
PACKAGEamsmath	250
PACKAGEbabel	58
PACKAGEbiblatex	56
PACKAGEbidi	63, 353
PACKAGEccaption	68



PACKAGEcolor	57
PACKAGEedmac	1, 5, 9–12, 59, 162, 168, 217, 245, 290, 299, 341
PACKAGEedstanza	1, 12, 244
PACKAGEeledmac	1, 9, 12–14, 48, 49, 112, 168, 237, 240, 254, 276, 287, 291, 293–295, 345, 347, 349
PACKAGEeledpar	69, 141, 274, 299, 344, 347–350
PACKAGEetex	353
PACKAGEtoolbox	73, 112, 196, 205, 212, 229, 274, 285
PACKAGEfootmisc	28, 58, 168, 299
PACKAGEhandout	348
PACKAGEhyperref	45, 218, 242, 279, 346–348
PACKAGEifluatex	61
PACKAGEifxetex	61
PACKAGEimakeidx	48, 56, 62, 237, 240, 294, 345–347, 349
PACKAGEindextool	294
PACKAGEindextools	48, 56, 62, 237, 240, 294, 349
PACKAGEinputenc	114
PACKAGEledarab	58
PACKAGEledmac	1, 9, 12, 58, 73, 240, 290, 291, 294, 297
PACKAGEledpar	58
PACKAGEMemoir	62, 240, 294, 299, 348
PACKAGEMorewrites	56
PACKAGEMusixtex	347
PACKAGEpolyglossia	34, 58, 108, 139, 159
PACKAGERagged2e	36, 61
PACKAGEledmac	1, 2, 9, 10, 12–15, 17, 18, 20–22, 24–26, 28, 31, 33, 35–40, 42, 43, 45, 46, 48–50, 54, 56–60, 74, 77, 82, 84, 86, 87, 90, 98, 105, 135, 142, 144, 149, 168, 189, 196, 199, 205, 212, 221, 224, 240, 254, 274, 275, 287, 294, 295, 297, 298, 352
PACKAGEReledpar	1, 3, 5, 7, 14, 16, 38, 54–56, 58, 60, 69, 75, 85, 90, 107, 142, 145, 184, 185, 196, 211, 213, 214, 236, 244, 353
PACKAGESuffix	61
PACKETabmac	1, 12, 299
PACKAGEuninormalize	26
PACKAGExargs	25, 61
PACKAGExkeyval	60, 211
PACKAGExstring	61, 242

## A

\absline@num	1
Abu Kamil Shuja' b. Aslam	11
\actionlines@list	1
\actions@list	1
\add@inserts	1
\add@inserts@next	1
\add@penalties	1
\addtol@denvbody	1
Adelard II	11
\advancelabel@refs	1
\advanceline	1, 20
\advancepageno	1
\Aendnote	22



<code>\affixline@num</code> .....	1
<code>\affixpstart@num</code> .....	1
<code>\affixside@note</code> .....	1
<code>\Afootnote</code> .....	22
<code>\afternoteX</code> .....	37
<code>\afterruleX</code> .....	38
<code>\ampersand</code> .....	1, 43
<code>\applabel</code> .....	1, 46
<code>\appref</code> .....	1, 46
<code>\apprefwithpage</code> .....	1, 46
<code>\arrangementX</code> .....	1, 29
<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, 16
<code>\AtEveryPstart</code> .....	1, 16
<code>\autopar</code> .....	1, 16

**B**

<code>\ballast</code> .....	56
<code>\ballast@count</code> .....	1
Beeton, Barbara Ann Neuhaus Friend .....	16
<code>\beforeeledchapter</code> .....	1
<code>\beforenotesX</code> .....	37
<code>\beginnumbering</code> .....	1, 14
<code>\Bendnote</code> .....	22
<code>\Bfootnote</code> .....	22
<code>\bhooknoteX</code> .....	36
<code>\bodyfootmarkA</code> .....	27
<code>\boxfootnotenumbers</code> .....	1
Bredon, Simon .....	11
Breger, Herbert .....	12, 253
Brey, Gerhard .....	11
Busard, Hubert L. L. ....	11
<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> .....	22
<code>\Cfootnote</code> .....	22



<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 .....	11
Claassens, Geert H. M. ....	11
<code>\colalignX</code> .....	36
Copernicus, Nicolaus .....	11
<code>\critext</code> .....	290
<code>\ctab</code> .....	1
<code>\ctabtext</code> .....	1

## D

Dekker, Dirk-Jan .....	57
<code>\Dendnote</code> .....	22
<code>\Dfootnote</code> .....	22
<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, 20
<code>\dolinehook</code> .....	1, 20
<code>\dosplits</code> .....	1
Downes, Michael .....	57, 148, 150
<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, 52, 267
<code>edarrayc</code> (environment) .....	50
<code>edarrayl</code> (environment) .....	50
<code>edarrayr</code> (environment) .....	50
<code>\edatleft</code> .....	1, 52



<code>\edatright</code>	1, 52
<code>\edbeforetab</code>	1, 52, 267
<code>\edfilldimen</code>	1
<code>\edfont@info</code>	1
<code>\EDINDEX</code>	1
<code>\edindex</code>	1, 48
<code>\edindexlab</code>	1, 49
<code>\EDLABEL</code>	1
<code>\edlabel</code>	1, 44
<code>\edlineref</code>	1, 44
<code>\edmakelabel</code>	1, 45
<code>\edpageref</code>	1, 44
<code>\edrowfill</code>	1, 51
<code>\EDTAB</code>	1
<code>\edtabcolsep</code>	1, 50
<code>\EDTABINDENT</code>	1
<code>\edtabindent</code>	1
<code>\EDTABtext</code>	1
<code>edtabularc (environment)</code>	50
<code>edtabularl (environment)</code>	50
<code>edtabularr (environment)</code>	50
<code>\EDTEXT</code>	1
<code>\edtext</code>	1, 21
<code>\edvertdots</code>	1, 52
<code>\edvertline</code>	1, 52
<code>\Eendnote</code>	22
<code>\Efootnote</code>	22
<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, 19
<code>\endminipage</code>	1
<code>\endnumbering</code>	1, 14
<code>\endpage@num</code>	1
<code>\endprint</code>	1
<code>\endquotation</code>	1



<code>\endquote</code> .....	1
<code>\endsub</code> .....	1, 19
<code>\endsubline@num</code> .....	1
environments:	
<code>edarrayc</code> .....	50
<code>edarrayl</code> .....	50
<code>edarrayr</code> .....	50
<code>edtabularc</code> .....	50
<code>edtabularl</code> .....	50
<code>edtabularr</code> .....	50
<code>ledgroup</code> .....	43
<code>ledgroupsize</code> .....	43
<code>minipage</code> .....	43
Euclid .....	11
<code>\extensionchars</code> .....	1, 55

## 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, 43
<code>\flush@notes</code> .....	1
<code>\fnpos</code> .....	1, 28
Folkerts, Menso .....	11
<code>\footfootmarkA</code> .....	27
<code>\footfudgefiddle</code> .....	1, 57
<code>\footnoteA</code> .....	27
<code>\footnoteB</code> .....	27
<code>\footnoteC</code> .....	27
<code>\footnoteD</code> .....	27
<code>\footnoteE</code> .....	27
<code>\footnotelang@lua</code> .....	1
<code>\footnotelang@poly</code> .....	1
<code>\footnoteoptions@</code> .....	1
<code>\footplitskips</code> .....	1
<code>\fullstop</code> .....	1, 39

## G

Gädeke, Nora .....	12
<code>\get@edindex@hyperref</code> .....	1
<code>\get@edindex@ledinnote@command</code> .....	1
<code>\get@index@command</code> .....	1



\get@linelistfile .....	1
\get@sw@txt .....	1
\getline@num .....	1
\gl@p .....	1

## H

\h@num .....	1
\hangindentX .....	36
\hidenumbering .....	1, 20
\Hilfsbox .....	1
\hilfsbox .....	1
\hilfscount .....	1
\HILFSskip .....	1
\Hilfsskip .....	1
\hilfsskip .....	1
\hsizethreecolX .....	37
\hsizetwocolX .....	37
\hyperlinkformat .....	1
\hyperlinkformatR .....	1
\hyperlinkR .....	1

## I

\if@addsw .....	1
\if@edindex@fornote@true .....	1
\if@eled@sectioning .....	1
\if@led@nofoot .....	1
\if@lemmacommand@ .....	1
\if@noeled@sec .....	1
\if@noneed@Footnote .....	1
\if@RTL .....	1
\ifautopar@pause .....	1
\ifbypage@ .....	1
\ifbypage@R .....	1
\ifbypstart@ .....	1
\ifbypstart@R .....	1
\ifeledmaccompat@ .....	1
\iffirst@linenum@out@ .....	1
\ifinserthangingsymbol .....	1
\ifistanza .....	1
\ifl@d@dash .....	1
\ifl@d@elin .....	1
\ifl@d@esl .....	1
\ifl@d@pnum .....	1
\ifl@d@ssub .....	1
\ifl@d@Xmorethantwolines .....	1
\ifl@d@Xtwolines .....	1
\ifl@dend@X .....	1
\ifl@dhidenumber .....	1
\ifl@dmemoir .....	1
\ifl@dpaging .....	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, 55
<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>\ifnoedgroup@</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>\insertthangingymbol</code> .....	1
<code>\insertlines@list</code> .....	1
<code>\inserts@list</code> .....	1

## J

Jayaditya .....	12
-----------------	----

## K

Kabelschacht, Alois .....	138
---------------------------	-----

## 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@ddodoreinxtrafeet</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@dfeetbeginmini</code>	1
<code>\l@dfeetendmini</code>	1
<code>\l@dgetline@margin</code>	1
<code>\l@dgetlock@disp</code>	1
<code>\l@dgetref@num</code>	1
<code>\l@dgetsidenote@margin</code>	1
<code>\l@dgonblearg</code>	1
<code>\l@dlabel@parse</code>	1
<code>\l@dld@ta</code>	1
<code>\l@dlp@rbox</code>	1
<code>\l@dlsn@te</code>	1
<code>\l@dlsnote</code>	1
<code>\l@dmake@labels</code>	1
<code>\l@dmodfordetext</code>	1
<code>\l@dnullfills</code>	1
<code>\l@dnumpstartsL</code>	1
<code>\l@doldold@footnotetext</code>	1
<code>\l@dp@rsefootspec</code>	1
<code>\l@dpagingfalse</code>	1
<code>\l@dpagingtrue</code>	1



<code>\l@dpairingfalse</code> .....	1
<code>\l@dpairingtrue</code> .....	1
<code>\l@dparsedendline</code> .....	1
<code>\l@dparsedendpage</code> .....	1
<code>\l@dparsedendsub</code> .....	1
<code>\l@dparsedstartline</code> .....	1
<code>\l@dparsedstartpage</code> .....	1
<code>\l@dparsedstartsub</code> .....	1
<code>\l@dparsefootspec</code> .....	1
<code>\l@dpprintingcolumnsfalse</code> .....	1
<code>\l@dpprintingcolumnstrue</code> .....	1
<code>\l@dpprintingpagesfalse</code> .....	1
<code>\l@dpprintingpagestrue</code> .....	1
<code>\l@dpush@begins</code> .....	1
<code>\l@drd@ta</code> .....	1
<code>\l@dref@undefined</code> .....	1
<code>\l@drestorefills</code> .....	1
<code>\l@dstoreforedtext</code> .....	1
<code>\l@drp@rbox</code> .....	1
<code>\l@drsn@te</code> .....	1
<code>\l@drsnote</code> .....	1
<code>\l@dsetmaxcolwidth</code> .....	1
<code>\l@dskipnumberfalse</code> .....	1
<code>\l@dskipnumbertrue</code> .....	1
<code>\l@dtabaddcols</code> .....	1
<code>\l@dtabnoexpands</code> .....	1
<code>\l@dunboxmpfoot</code> .....	1
<code>\l@dunhbox@line</code> .....	1
<code>\l@dzeropenalties</code> .....	1
<code>\label</code> .....	46
<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 .....	10
<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@@docclearpage</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@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 (environment)</code> .....	43
<code>ledgroupsize (environment)</code> .....	43
<code>\ledinnernote</code> .....	1, 47
<code>\ledinnote</code> .....	1
<code>\ledinnotehyperpage</code> .....	1
<code>\ledinnotemark</code> .....	1



<code>\ledleftnote</code>	1, 47
<code>\ledlinenum</code>	1
<code>\ledllfill</code>	1
<code>\ledlsnotefontsetup</code>	1, 47
<code>\ledlsnotesep</code>	1, 47
<code>\ledlsnotewidth</code>	1, 47
<code>\lednopb</code>	1, 54
<code>\lednopbinversetrue</code>	55
<code>\lednopbnum</code>	1
<code>\ledouternote</code>	47
<code>\ledouterote</code>	1
<code>\ledpb</code>	1, 54
<code>\ledpbnum</code>	1
<code>\ledpbsetting</code>	1, 55
<code>\ledrightnote</code>	1, 47
<code>\ledrlfill</code>	1
<code>\ledrsnotefontsetup</code>	1, 47
<code>\ledrsnotesep</code>	1, 47
<code>\ledrsnotewidth</code>	1, 47
<code>\ledsectnomark</code>	1
<code>\ledsectnotoc</code>	1
<code>\ledsetnormalparstuff@common</code>	1
<code>\ledsetnormalparstuffX</code>	1
<code>\ledsidenote</code>	1, 47
<code>\leftctab</code>	1
<code>\leftlinenum</code>	1, 19
<code>\leftltab</code>	1
<code>\leftnoteupfalse</code>	47
<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, 18
<code>\linenum</code>	1, 24
<code>\linenum@out</code>	1
<code>\linenumberlist</code>	1, 18
<code>\linenumberstyle</code>	1, 20
<code>\linenumincrement</code>	1, 18
<code>\linenummargin</code>	1, 18
<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, 19
Lorch, Richard	11
<code>\ltab</code>	1
<code>\ltabtext</code>	1
Luecking, Dan	61

## M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\manage stanza@modulo</code>	1
<code>\maxhnotesX</code>	38
Mayer, Gyula	12
<code>\measurebody</code>	1
<code>\measuremcell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
<code>\measuretbody</code>	1
Middleton, Thomas	12, 81
<code>minipage</code> (environment)	43
Mittelbach, Frank	11
<code>\morenoexpands</code>	1, 57
<code>\mpfnpos</code>	1, 28
<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>\new@line</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> .....	54
<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> .....	35
<code>\notenumfontX</code> .....	35
<code>\noteschanged@false</code> .....	1
<code>\noteschanged@true</code> .....	1
<code>\noteswidthliketwocolumnsX</code> .....	38
<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, 16
<code>\numberpstarttrue</code> .....	1, 16
<code>\numberstanzafalse</code> .....	42
<code>\numberstanzatrue</code> .....	42



\numlabfont ..... 1, 39

## O

\old@hsize ..... 1  
 \one@line ..... 1  
 optionlinrangesep ..... 211

## P

\page@action ..... 1  
 \page@num ..... 1  
 \pagelinesep ..... 1, 48  
 \pageno ..... 1  
 \pageref ..... 46  
 \par@line ..... 1  
 \para@footgroupX ..... 1  
 \para@footsetup ..... 1  
 \para@footsetupX ..... 1  
 \para@vfootnoteX ..... 1  
 \parafootfmt ..... 1  
 \parafootfmtX ..... 1  
 \parafootgroup ..... 1  
 \parafootsepX ..... 37  
 \parafootstart ..... 1  
 \parafootstartX ..... 1  
 \paravfootnote ..... 1  
 \parindentX ..... 35  
 \pausenumbering ..... 1, 17  
 \pend ..... 1, 15  
 Plato of Tivoli ..... 11  
 \postbodyfootmark ..... 1  
 \prebodyfootmark ..... 1  
 \prenotesX ..... 38  
 \prepare@edindex@fornote ..... 1  
 \prepare@prenotesX ..... 1  
 \prepare@preXnotes ..... 1  
 \prev@nopb ..... 1  
 \prev@pb ..... 1  
 \prevpage@num ..... 1  
 \preXnotes ..... 1, 38  
 \preXnotes@ ..... 1  
 \print@eledsection ..... 1  
 \print@footnoteXrule ..... 1  
 \print@leftmargin@eledsection ..... 1  
 \print@line ..... 1  
 \print@notesX ..... 1  
 \print@rightmargin@eledsection ..... 1  
 \print@Xfootnoterule ..... 1  
 \print@Xnotes ..... 1  
 \printendlines ..... 1  
 \printlineendnote ..... 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>\pst@rtedLfalse</code> .....	1
<code>\pst@rtedLtrue</code> .....	1
<code>\pstart</code> .....	1, 15
<code>\pstarteref</code> .....	1
<code>\pstartnum</code> .....	1
<code>\pstartref</code> .....	44

## Q

<code>\quotation</code> .....	1
<code>\quote</code> .....	1

## R

<code>\raggedX</code> .....	37
<code>\raw@text</code> .....	1
<code>\rbracket</code> .....	1
<code>\read@linelist</code> .....	1
<code>\ref</code> .....	46
<code>\Relax</code> .....	1
<code>\reledmac@error</code> .....	1
<code>\reledmac@warning</code> .....	1
<code>\removehboxes</code> .....	1
<code>\resetprevline@</code> .....	1, 84
<code>\resetprevpage@</code> .....	1
<code>\resetprevpage@num</code> .....	85
<code>\restore@familiarnotes</code> .....	1
<code>\restore@notes</code> .....	1
<code>\restore@sidenotes</code> .....	1
<code>\resumenumbering</code> .....	1, 17
<code>\rightctab</code> .....	1
<code>\rightlinenum</code> .....	1, 19
<code>\rightltab</code> .....	1
<code>\rightnoteupfalse</code> .....	47
<code>\rightrtab</code> .....	1
<code>\rightstartnum</code> .....	1
<code>\rigidbalance</code> .....	1
<code>\rtab</code> .....	1
<code>\rtabtext</code> .....	1



## S

Sacrobosco .....	12
\sameword .....	<u>1</u> , 25
\sameword@inedtext .....	<u>1</u>
Schöpf, Rainer .....	11
\section@num .....	<u>1</u>
\select@lemmafонт .....	<u>1</u>
\select@lemmafонт .....	<u>1</u> , 39
\series .....	<u>1</u>
\seriesatbegin .....	<u>1</u> , 28
\seriesatend .....	<u>1</u> , 28
\set@line .....	<u>1</u>
\set@line@action .....	<u>1</u>
\setapprefprefixmore .....	<u>1</u> , 46
\setapprefprefixsingle .....	<u>1</u> , 46
\setcommand@series .....	<u>1</u>
\sethangingsymbol .....	<u>1</u> , 42
\setistwofollowinglines .....	<u>1</u>
\setl@dlp@rbox .....	<u>1</u>
\setl@drpr@box .....	<u>1</u>
\setline .....	<u>1</u> , 20
\setlinenum .....	<u>1</u> , 20
\setmcellcenter .....	<u>1</u>
\setmcellleft .....	<u>1</u>
\setmcellright .....	<u>1</u>
\setmrowcenter .....	<u>1</u>
\setmrowleft .....	<u>1</u>
\setmrowright .....	<u>1</u>
\setnoteswidthliketwocolumnsX@ .....	<u>1</u>
\setnotesXpositionliketwocolumns@ .....	<u>1</u>
\setprintendlines .....	<u>1</u>
\setprintlines .....	<u>1</u>
\setsidenotessep .....	48
\setstanzaindents .....	<u>1</u> , 40
\setstanzapenalties .....	<u>1</u> , 41
\setstanzavalues .....	<u>1</u>
\settccllcenter .....	<u>1</u>
\settccllleft .....	<u>1</u>
\settccllright .....	<u>1</u>
\settoggle@series .....	<u>1</u>
\setthrowcenter .....	<u>1</u>
\setthrowleft .....	<u>1</u>
\setthrowright .....	<u>1</u>
\setXnotespositionliketwocolumns@ .....	<u>1</u>
\setXnoteswidthliketwocolumns@ .....	<u>1</u>
\showlemma .....	<u>1</u> , 55
\showwordrank .....	<u>1</u> , 27
\sidenote@margin .....	<u>1</u>
\sidenotemargin .....	<u>1</u> , 47
\sidepstartnumtrue .....	17



<code>\skip@lockoff</code> .....	1
<code>\skipnumbering</code> .....	1, 20
<code>\splitoff</code> .....	1
<code>\spreadmath</code> .....	1, 51
<code>\spreadtext</code> .....	1, 51
<code>\stanza</code> .....	1, 40
<code>\stanza@count</code> .....	1
<code>\stanza@hang</code> .....	1
<code>\stanza@line</code> .....	1
<code>\stanzaindent</code> .....	1, 41
<code>\stanzaindent*</code> .....	1, 41
<code>\stanzaindentbase</code> .....	1, 40
<code>\stanzanumwrapper</code> .....	1, 43
<code>\startlock</code> .....	1, 19
<code>\startsub</code> .....	1, 19
<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, 20
<code>\sublinenumincrement</code> .....	1, 18
<code>\sublinenumr@p</code> .....	1
<code>\sublinenumrep</code> .....	1
<code>\sublineref</code> .....	1, 44
<code>\sublines@false</code> .....	1
<code>\sublines@true</code> .....	1
<code>\sublock@disp</code> .....	1
<code>\sublockdisp</code> .....	1
Sullivan, Wayne .....	11, 12, 56, 68, 72, 148, 149, 217, 244
<code>\sza@penalty</code> .....	1

## T

<code>\tabHilfbox</code> .....	1
<code>\tabhilfbox</code> .....	1
<code>\theadcolcount</code> .....	1
<code>\theendpageline</code> .....	1
<code>\thefootnoteA</code> .....	27
Theodosius .....	12
<code>\thepageline</code> .....	1
<code>\thepstart</code> .....	1, 16
<code>\thestanza</code> .....	1, 42
<code>\thestartpageline</code> .....	1
<code>\this@line@list@version</code> .....	1
<code>\threecolfootfmt</code> .....	1
<code>\threecolfootfmtX</code> .....	1
<code>\threecolfootgroup</code> .....	1
<code>\threecolfootgroupX</code> .....	1



<code>\threecolfootsetup</code> .....	1
<code>\threecolfootsetupX</code> .....	1
<code>\threecolvfootnote</code> .....	1
<code>\threecolvfootnoteX</code> .....	1
<code>\twocolfootfmt</code> .....	1
<code>\twocolfootfmtX</code> .....	1
<code>\twocolfootgroup</code> .....	1
<code>\twocolfootgroupX</code> .....	1
<code>\twocolfootsetup</code> .....	1
<code>\twocolfootsetupX</code> .....	1
<code>\twocolvfootnote</code> .....	1
<code>\twocolvfootnoteX</code> .....	1

## U

<code>\unvxhX</code> .....	1
----------------------------	---

## V

Vamana .....	12
<code>\variab</code> .....	1
<code>\vbfnoteX</code> .....	1
<code>\vl@dbfnote</code> .....	1
<code>\vl@dcsnote</code> .....	1
<code>\vl@dlsnote</code> .....	1
<code>\vl@drsnote</code> .....	1
<code>\vnumfootnoteX</code> .....	1

## W

Whitney, Ron .....	11
<code>\wrap@edcrossref</code> .....	1
Wujastyk, Dominik .....	10

## X

<code>\X@doreinfeet</code> .....	1
<code>\Xafterlemmaseparator</code> .....	34
<code>\Xafternote</code> .....	37
<code>\Xafternumber</code> .....	32
<code>\Xafterrule</code> .....	38
<code>\Xaftersymlinenum</code> .....	32
<code>\Xarrangement</code> .....	1, 29
<code>\Xarrangement@normal</code> .....	1
<code>\Xarrangement@paragraph</code> .....	1
<code>\Xarrangement@threecol</code> .....	1
<code>\Xarrangement@twocol</code> .....	1
<code>\Xbeforelemmaseparator</code> .....	34
<code>\Xbeforenotes</code> .....	37
<code>\Xbeforenumber</code> .....	32
<code>\Xbeforesymlinenum</code> .....	32
<code>\Xbhooknote</code> .....	36
<code>\Xboxlinenum</code> .....	33
<code>\Xboxlinenumalign</code> .....	33



\Xboxsymlinenum	33
\Xcolalign	36
\Xdo@feet	<u>1</u>
\xedindex	<u>1</u>
\xedlabel	<u>1</u>
\xedtext	<u>1</u>
\Xendafterenumber	32
\Xendafterlemmaseparator	34
\Xendafternote	39
\Xendaftersymlinenum	32
\Xendahookinplaceofnumber	34
\Xendahooklinenumber	34
\Xendbeforelemmaseparator	34
\Xendbeforenumber	32
\Xendbeforesymlinenum	32
\Xendbhookinplaceofnumber	34
\Xendbhooklinenumber	34
\Xendbhooknote	36
\Xendboxendlinenumalign	33
\Xendboxlinenum	33
\Xendboxlinenumalign	33
\Xendboxstartlinenumalign	33
\Xendboxsymlinenum	33
\Xendhangindent	36
\Xendinplaceoflemmaseparator	35
\Xendinplaceofnumber	33
\Xendlemmadisablefontselection	35
\Xendlemmaseparator	34
\Xendlinerrangeseparator	30
\Xendmorethantwolines	31
\Xendmorethantwolinesapprefwithpage	46
\Xendnonumber	31
\Xendnotefontsize	35
\Xendnotenumfont	35
\Xendnumberonlyfirstinline	30
\Xendnumberonlyfirstintwolines	30
\Xendparagraph	39
\Xendsep	39
\Xendsymlinenum	30
\Xendtwolines	31
\Xendtwolinesapprefwithpage	46
\Xendtwolinesbutnotmore	31
\Xendtwolinesbutnotmoreapprefwithpage	46
\Xendtwolinesonlyinsamepageapprefwithpage	46
\Xhangindent	36
\Xhsizethreecol	37
\Xhsizetwocol	37
\Xinplaceoflemmaseparator	34
\Xinplaceofnumber	33
\Xinsertparafootsep	<u>1</u>



\Xledsetnormalparstuff .....	1
\xleft@appenditem .....	1
\Xlemmadisablefontselection .....	35
\Xlemmaseparator .....	34
\Xlinangeseparator .....	30
\xlineref .....	1, 45
\Xmaxhnotes .....	38
\Xmorethantwolines .....	31
\Xmorethantwolinesappref .....	46
\Xnolemmaseparator .....	1, 34
\Xnonbreakableafternumber .....	32
\Xnonumber .....	31
\Xnotefontsize .....	35
\Xnotenumfont .....	35
\Xnoteswidthliketwocolumns .....	38
\Xnumberonlyfirstinline .....	30
\Xnumberonlyfirstintwolines .....	30
\Xonlypstart .....	32
\xpageref .....	1, 45
\Xparafootsep .....	37
\Xparindent .....	35
\Xpstart .....	31
\Xpstarteverytime .....	32
\xpstartref .....	1, 45
\Xragged .....	37
\xright@appenditem .....	1
\Xstanza .....	32
\Xstanzaseparator .....	32
\xsublineref .....	1, 45
\Xsymlinenum .....	30
\Xtwolines .....	31
\Xtwolinesappref .....	46
\Xtwolinesbutnotmoreappref .....	46
\Xtwolinesonlyinsamepage .....	31, 46
\Xtxtbeforenotes .....	37
\Xunvxh .....	1
\xxref .....	1, 45

## Z

\zz@@@ .....	1
--------------	---



## Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added tabmac code, and extended indexing	1
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	62
\morenoexpands: Added \l@dtabnoexpands to \no@expands	106
\reledmac@error: Added \eledmac@error and replaced error messages	63
v0.2.1.	
\@lab: Removed page setting from \@lab	220
General: Added text about normal labeling	46
Bug fixes and match with mempatch v1.8	1
Major changes to insert code when memoir is loaded	215
\dextrafeet: Renamed \dextrafeet to \l@ddextrafeet	212
\edlabel: Tweaked \edlabel to get correct page numbers	218
\l@ddodoreinxtrafeet: Renamed \dodoreinxtrafeet to \l@ddodoreinxtrafeet	213
\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	106
\zz@@@: Minor change to \zz@@@	217
v0.2.2.	
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts pack- age	68
\footfudgefiddle: Added \footfudgefiddle	147
\line@list@stuff: Added initial write of page number in \line@list@stuff	99
\para@footsetup: Added \footfudgefiddle to \para@footsetup	147
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	180
v0.3.0.	
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	220
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	88
General: Includes edstanza and more	1
\ledlinenum: Added \linenumr@p and \sublinenumr@p to \leftlinenum and \rightlinenum	80
\linenumberlist: Added \linenumberlist mechanism	68
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	193
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	167
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	79
v0.3.1.	
General: Not released. Added remarks about the parallel package	1
v0.4.0.	
\@iiiminipage: Modified kernel \@iiiminipage and \endminipage to cater for criti- cal footnotes	234
General: Added final/draft options	60
Added ledgroup environment	235
Added ledgroupsize environment	235
Added minipage, etc., support	1



\edtext: Added \showlemma to \edtext	106
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	232
\mpnormalfootgroup: Added \mpnormalfootgroup	145
\mpnormalvfootnote: Added \mpnormalvfootnote	143
\showlemma: Added \showlemma	68
\Xarrangement@normal: Added minpage footnote setup to \footnormal	142
v0.4.1.	
General: Added code for changing \@docclearpage	215
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	168
\edindex: Let eledmac take advantage of memoir's indexing	240
\print@Xnotes: Added \@opXfeet	213
\Xdo@feet: Changed \Xdo@feet code for easier extensions	212
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	169
\@xympar: Eliminated \marginpar disturbance	226
General: Added left and right side notes	226
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
General: Added moveable side note	226
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place	235
\affixline@num: Changed \affixline@num to cater for sidenotes	130
\l@dgetsidenote@margin: Added \sidenotemargin and \sidenote@margin	227
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	90
\@nl@reg: Added \fix@page to \@nl	88
Extended \@nl to include the page number	88
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
\fix@page: Added \last@page@num and \fix@page	89
\new@line: Extended \new@line to output page numbers	99
\l@dbfnote: Changed \l@dbfnote and \l@dbfnote as originals could give incorrect markers in the footnotes	169
v0.7.0.	
\@nl@reg: Added \@nl@reg	88
\@ref@reg: Added \@ref@reg	96
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	63
Replaced all \interfootnotelinepenalty, etc., by just \interfootnotelinepenalty	1
Tidying up for eledpar and ledarab packages	1
\affixline@num: Added skipnumering to \affixline@num	130
\do@actions@fixedcode: Added \do@actions@fixedcode	129



\do@actions@next: Added number skipping to \do@actions	128
\do@insidelinehook: Added \do@linehook for use in \do@line	126
\endnumbering: Changed \endnumbering for eledpar	71
\fx@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \fx@l@cks	132
\footsplitskips: Added \footsplitskips for use in many footnote styles	141
\get@linelistfile: Added \get@linelistfile	86
\ifledRcol@: Added \l@dunpstartsL, \ifl@dpairing and \ifpst@rted for/from eledpar	69
\initnumbering@reg: Added \initnumbering@reg	70
\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot containing some common code	234
\l@dcsnotetext@r: Added \l@demptyd@ta	126
\l@dgetline@margin: Added \l@dgetline@margin	76
\l@dgetlock@disp: Added \l@dgetlock@disp	78
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	227
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	126
\l@dzeropenalties: Added \l@dzeropenalties	121
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	80
\line@list@stuff: Deleted \page@start from \line@list@stuff	99
\list@clearing@reg: Added \list@clearing@reg	86
\n@num: Added \n@num	95
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	174
\resumenumbering: Changed \resumenumbering for eledpar	72
\setprintendlines: Added \setprintendlines for use by \printendlines	192
\setprintlines: Added \setprintlines for use by \printlines	164
\skipnumbering: Added \skipnumbering and supports	102
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	77
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	79
Using \sublinenumrep instead of \sublinenumr@p	79
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	174
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 us 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 eledpar, possibility to use \autopar on the right side.	1
Possibility to number \pstart.	16
Possibility to number the pstart with the commands \numberpstarttrue.	1



\ifledRcol@: Added \ifledRcol and \ifnumberingR for/from eledpar . . . . .	69
v0.12.1.	
General: Don't number \pstarts of stanza. . . . .	1
The numbering of \pstarts restarts on each \beginnumbering. . . . .	1
v0.13.0.	
General: New stanzaindentsrepetition counter to repeat stanza indents every $n$ verses. . . . .	40
New stanzaindentsrepetition counter: to repeat stanza indents every $n$ verses. . . . .	1
\managestanza@modulo: New stanzaindentsrepetition counter to repeat stanza indents every $n$ verses. . . . .	246
v0.13.1.	
General: \thepstartL and \thepstartR use now \bfseries and not \bf, which is deprecated and makes conflicts with memoir class. . . . .	1
v0.14.0.	
General: Tweaked \edlabel to get correct line number if the command is first element of a paragraph. . . . .	1
\edlabel: Tweaked \edlabel to get correct line number if the command is first element of a paragraph. . . . .	218
v0.15.0.	
General: Line numbering can be reset at each pstart. . . . .	74
Possibility to print \pstart number inside. . . . .	17
\affixline@num: Line numbering can be disabled. . . . .	130
\ifinserthangingsymbol: New management of hangingsymbol insertion, preventing undesirable insertions. . . . .	244
\printlines: Line numbering can be reset at each pstart. . . . .	166
v0.17.0.	
\ifinserthangingsymbol: New new management of hangingsymbol insertion, preventing undesirable insertions. . . . .	244
v1.0.0.	
General: \lemma can contain commands. . . . .	24
Debug in lineation command . . . . .	18
New generic commands to customize footnote display. . . . .	29, 204
Options nonum and noseup in \Xfootnote. . . . .	22
Options of \Xfootnotes. . . . .	138
Possibility to have commands in sidenotes. . . . .	47
Some compatibility break with eledmac. Change of name: eledmac. . . . .	1
\morenoexpands: Change to be compatible with new features . . . . .	106
v1.0.1.	
General: Correction on \Xnumberonlyfirstinline with lineation by pstart or by page. 30	
v1.1.0.	
General: Add \labelpstarttrue. . . . .	17
Add \Xnumberonlyfirstintwolines . . . . .	30
Add \Xpstart and \Xonlypstart . . . . .	31
New hook to add arbitrary code at the beginning of the notes . . . . .	36
New options for block of notes. . . . .	37
New package option: parapparatus. . . . .	1
New tools to change order of series . . . . .	203
Sectioning commands. . . . .	53
\preXnotes: New skip \preXnotes@ . . . . .	187
\settoggle@series: \settoggle@series switch the global value of the toggle, not only the local value. . . . .	205



v1.2.0.	
\endquote:	Compatibility of \ledchapter with the <i>memoir</i> class. . . . . 272
\preXnotes:	Debug in familiar footnotes (bug introduced by v1.1). . . . . 187
v1.3.0.	
\endquote:	<i>Quotation</i> and quote environment inside numbered sections. . . . . 272
v1.4.0.	
General:	Compatibility with LuaTeX of RTL notes. . . . . 1
\edtext:	Compatibility of \edtext with the right-to-left direction (with Polyglossia). 106
\ledsetnormalparstuffX:	Direction of footnotes with polyglossia. . . . . 183
\newseries@:	Remembers the language of the lemma, in order to create a correct direction for the footnote separator. . . . . 198
\rbracket:	Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). . . . . 159
v1.4.1.	
\affixside@note:	Remove spurious spaces. . . . . 231
\endquote:	New option <i>noquotation</i> . . . . . 272
\labelrefsparsesubline:	Fix bug with \edlabel. . . . . 218
\vl@dbfnote:	Compatibility of standard footnotes with eledmac when these footnotes contain any commands. . . . . 169
v1.4.2.	
General:	Debug with some special classes. . . . . 1
v1.4.3.	
General:	Add \Xnonbreakableafternumber. . . . . 32
	Spurious space after familiar footnotes. . . . . 1
v1.4.4.	
General:	Label inside familiar footnotes. . . . . 1
v1.4.5.	
General:	Bug with komasscript + eledpar + chapter. . . . . 1
v1.4.6.	
General:	Bug with memoir class introduced by 1.4.5. . . . . 1
v1.4.7.	
\endquote:	Compatibility of sectioning commands with \autopar. . . . . 272
v1.4.8.	
General:	Corrects a bug with parallel texts introduced by 1.1. . . . . 1
v1.4.9.	
\normalbfnoteX:	Allow to redefine \thefootnoteX with alph when some packages are loaded. . . . . 174
v1.5.0.	
General:	Correct indexing when the call is made in critical notes. . . . . 237
\do@insidelinehook:	Added \do@insidelinehook for use in \do@line . . . . . 126
\edindex:	Compatibility with imakeidx package, and possibility to use multiple index with \edindex. . . . . 240
v1.5.1.	
\managestanza@modulo:	Correct stanzaindentsrepetition counter . . . . . 246
\normalvfootnoteX:	Fix bug with normal familiar footnotes when mixing RTL and LTR text. . . . . 171
v1.6.0.	
\newverse:	Add \falseverse macro. . . . . 248
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. . . . .	118
\ifinserthangingsymbol: Hang verse is now not automatically flush right. . . . .	244
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill in-side. . . . .	123
\pend: Spurious space in \pend. . . . .	120
v1.7.0.	
General: New features for managing page breaks. . . . .	54
v1.8.0.	
General: Compatibility with parledgroup option of eledpar package. . . . .	1
If imakeidx and hyperref are loaded, adds hyperref in the index. . . . .	237
\endquote: Correction of sectioning commands in parallel texts. . . . .	272
\get@index@command: Debug \get@index@command and compatibility with hyperref package. . . . .	239
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work. . . . .	208
\prevpage@num: Correct \parafootsep when using with ledgroup. . . . .	152
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined). . . . .	188
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, with eledpar. . . . .	237
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes. . . . .	212
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / led-group. . . . .	232
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. . . . .	272
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. . . . .	35
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. ....	88
General: Add \ledinnernote and \ledouternote commands. ....	47
Add \Xendparagraph and related settings. ....	39
Add hyperlink to crossref (needs hyperref package). ....	44
Compatibility with musixtex. ....	1
Debug eledmac sectioning command after using \resumenumbering. ....	1
Ensure that imakeidx is loaded <i>before</i> eledmac ....	237
New hooks: \Xafterrule and \afterruleX ....	38
New options for ragged-paragraph notes ....	37
New sectioning commands. ....	53
Optional arguments for \pstart and \pend. ....	16
\AtEveryPstart: New optional argument for \pstart, to execute code before it. ...	118
\edindex: Use correctly default index when imakeidx is loaded. ....	240
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands. ....	272
\ifledRcol@: Add \ifledRcol@ for eledpar ....	69
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar. ....	70
\l@dcnote: \l@dlsnote, \l@drsnote and \l@dcnote defined only one time, in eledmac, including needs for eledpar case. ....	228
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar. ....	227
\l@dunhbox@line: \do@line is split in more little commands. ....	124
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX. ....	208
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph. ....	208
\pend: New optional argument for \pend, to execute code after it. ....	120
\stanza: &can have an optional argument: content to be printed after. ....	248
\Stanza can have an optional argument: content to be printed before. ....	248
Add \newverse macro, \falseverse deprecated. ....	248

## v1.12.1.

\wrap@edcrossref: Fix spurious spaces. ....	221
---	-----

## v1.12.2.

\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0) ....	123
---	-----

## v1.12.3.

General: Add macros for new messages since v0.7 ....	63
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. ....	47
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 ....	100
\flag@start send a error message when a \edtext is done without insert (note) ....	100
\reledmac@error: Replaced error messages ....	63

## 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	242
\@ssect: Debug \eledchapter in parallel with memoir	276
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	126
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	71
\l@dgobblearg: \l@dgobblearg becomes \l@dgobbeloptarg	256
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	256
\resumenumbering: Debug \resumenumbering	72
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	221
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	100
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	38
Added widthliketwocolumns option	60
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	207
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	180
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	205
v1.13.1.	
General: Coming back of page and line breaking penalties's management, deleted by error in v0.17.	1
Debug quotation environment inside of a \pstart preceded by a sectioning command.	1
\thepstart: Add \l@dzeropenalties in \pstart	119
v1.13.2.	
General: Fix bug with normal footnotes, added by v1.13.0.	1
\ifledRcol@: Add \ifl@dpadding for eledpar	69
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.	188
Delete previous override of \l@d@@wrindexhyp at the beginning of a document when hyperref is not loaded.	243
Move gobbling command	68
Provide \@gobblefour	68
\edindex: Let eledmac take advantage of imakeidx even when memoir class is used	240
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref package.	279
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	276
v1.15.0.	
\@edtext@level: New boolean \if@edtext@.	106
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.	16



New tools to prevent ambiguous references in lemma	25
<code>\arrangementX@threecol</code> : Correct bug with paragraphed familiar footnotes setting.	180
<code>\endsub</code> : Restore subline feature (disabled by mistake in v1.8.0).	101
<code>\if@lemmacommand@</code> : New boolean <code>\iflemmacommand@</code> .	111
v1.15.1.	
<code>\line@list@stuff</code> : Revert modification of 1.5.2 which makes bug with numbering.	
Leave vertical mode to solve spurious space before minipage.	99
v1.16.0.	
General: <code>\edtext</code> is now defined only in <code>eledmac</code> , not in <code>eledpar</code> . Debug wrong numbering when using <code>\sameword + eledpar + \tag</code> command.	106
Compatibility of standard footnotes with some biblatex styles.	1
New <code>\stanzaindent</code> command.	1
v1.16.1.	
<code>\edlineref</code> : <code>\lineref</code> is not defined if defined by some other package, like <code>lineno</code> .	
Eledmac provides <code>\edlineref</code> instead.	222
v1.17.0.	
<code>\edtext</code> : Error message when calling <code>\edtext</code> outside of a numbered paragraph.	106
v1.18.0.	
<code>@edindex@hyperref</code> : Fix spurious space with <code>\edindex</code> when using <code>imakeidx/indextools + hyperref</code> .	242
General: Add <code>\Xstarteverytime</code>	31
Compatibility with Lua $\TeX$ RTL languages.	1
Debug <code>\Xonlypstart</code> when using <code>\Xnumberonlyfirstinline</code> and the current line number differs from the previous.	31
<code>\edlabel</code> : <code>\edlabel</code> is now defined only one time for both <code>eledmac</code> and <code>eledpar</code>	218
<code>\ifledRcol@</code> : Add <code>\ifl@dprintingpages</code> and <code>\@dprintingcolumns</code> for <code>eledpar</code>	69
<code>\l@d@section</code> : Option <code>parapparatus</code> works for endnotes.	189
<code>\print@line</code> : Compatibility with Lua $\TeX$ RTL languages.	124
<code>\printlinefootnote</code> : Code refactoring in <code>\printlinefootnote</code> : the printing of the numbers are factorized in <code>\printlinefootnotearea</code>	159
<code>\printpstart</code> : Debug <code>\Xpstart</code> with parallel pages and columns ( <code>eledpar</code> )	159
v1.19.0.	
General: <code>\Xmaxhnotes</code> and <code>\maxhnotesX</code> work now for both two-columns and three-columns setting.	1
Compatibility with <code>eledpar</code> v1.13.0.	1
<code>\footplitskips</code> : <code>\footplitskips</code> doesn't set <code>\floatingpenalty</code> to <code>\@MM</code> when processing parallel pages.	141
<code>\xxref</code> : <code>\xxref</code> works also with right side numbers, when <code>\@Rlineflag</code> is not empty.	223
v1.19.1.	
General: Call <code>\correct@footinsX@box</code> and <code>\correct@Xfootins@box</code> directly in <code>\print@notesX@forpages</code> and <code>\print@Xnotes@forpages</code> , that is in <code>eledpar</code> .	1
v1.20.0.	
General: Add <code>\Xendboxlinenum</code>	33
Add <code>\Xtwolines</code> and <code>\Xmorethantwolines</code> hooks	31
Add series option.	1
Correct <code>\Xinplaceofnumber</code> hook.	1
Explicit error message when calling <code>\Xfootnote</code> outside of <code>\edtext</code> .	1
Fix bug with line number typesetting direction when using <code>\eledsection</code> and similar commands for RTL texts with Lua $\TeX$ .	1
Fix issues with RTL text in notes when using Lua $\TeX$ .	1



Options fulllines in <code>\Xfootnote</code> .	22
The <code>\newifs</code> are not followed by boolean values set to false, because it is the $\TeX$ default setting.	1
<code>\printlines</code> : Added <code>\ifl@d@Xmorethantwolines</code> and <code>\ifl@d@Xmorethantwolines</code> to <code>\printlines</code>	167
<code>\stanza</code> : <code>&amp;</code> and <code>\&amp;</code> can be preceded by spaces.	248
<code>\xxref</code> : Debug <code>\xxref</code> when not loading <code>eledpar</code> (fix bug added in 1.19.0).	223
v1.21.0.	
<code>\@edindex@hyperref</code> : Look at the hyperindex option of hyperref before inserting hyperref	242
General: <code>\AtEveryPstart</code> and <code>\AtEveryPend</code> are now compatible with <code>\autopar</code>	1
<code>\Xafterrule</code> and <code>\afterruleX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\chapter</code> inside optional argument of <code>\pstart</code> works when typesetting parallel pages	1
<code>\preXnotes</code> and <code>\prenotesX</code> features no longer create problems of overflowing at the bottom of the page.	1
<code>\seriesatbegin</code> and <code>\seriesatbegin</code> more efficient	203
Add <code>\applabel</code> and related	46
Add <code>\beforenotesX</code> and <code>\Xbeforenotes</code> features for notes set in two and three column.	1
Add <code>\hidenumbering</code>	20
Add <code>\Xcolalign</code> and <code>\colalignX</code> hooks	36
Add <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> .	31
Add <code>\Xparindent</code> and <code>\hangindentX</code>	35
Add <code>\Xtwolinesbutnotmore</code> and <code>\Xtwolinesonlyinsamepage</code> .	1
Add <code>nocritical</code> , <code>noend</code> , <code>nofamiliar</code> and <code>noledgroup</code> options.	1
Add <code>noeledsec</code> package option	1
Debug <code>\beforenotesX</code> <code>\maxhnotesX</code> <code>\noteswidthliketwocolumnsX</code> and <code>\afterruleX</code> with footnotes set in two and three columns.	1
Fix bug when a <code>\Xfootnote</code> follows a <code>\Xendnote</code> in the second argument of <code>\edtext</code> (bug added in <code>eledmac</code> 1.0.0).	1
Fix bug with <code>\maxhnotesX</code> when using <code>\foottwocolX</code> or <code>\footthreecolX</code> .	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 <code>\doendnotes</code> . <code>oldprintnnumspace</code> option allows to come back to previous setting	1
<code>parapparatus</code> option works now with familiar footnotes.	1
Provide <code>\@gobblefive</code>	68
<code>\l@d@section</code> : <code>\endnotes</code> take five arguments.	189
<code>\ledinnotemark</code> : Add <code>\ledinnotemark</code> .	239
<code>\ledsetnormalparstuffX</code> : <code>\ledsetnormalparstuff</code> is deprecated and becomes <code>\ledsetnormalparstuffX</code> and <code>\Xledsetnormalparstuff</code> .	183
<code>\n@num</code> : <code>\n@num@ref</code> deleted	95
<code>\n@num</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	95
<code>\newhookcommand@series</code> : <code>\newhookcommand@series</code> can take an optional argument.	207
<code>\newhooktoggle@series</code> : <code>\newhooktoggle@series</code> can take an optional argument.	207
<code>\print@footnoteXrule</code> : Code refactoring: the spaces after the footnote rules are directly managed in <code>\print@Xfootnoterule</code> and <code>\print@footnoteXrule</code>	185



<code>\seriesatend</code> : Fix spurious space in <code>\seriesatend</code> . . . . .	204
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code> . . . . . .	102
Correct <code>\skipnumbering</code> for stanza. . . . .	102
Delete <code>\skipnumbering@reg</code> . . . . .	102
v1.22.0.	
General: Add <code>\doendnotesbysection</code> command. . . . .	23
Add option for lemma separator inside endnotes . . . . .	34
Adds hyperlink for references to notes in indices. . . . .	1
Fix conflict between <code>noend</code> package option and <code>edtabularx</code> environments . . . . .	1
Provides support for <code>xindy</code> . . . . .	1
Standardize endnotes handbook. . . . .	23
When using <code>hyperref</code> package, internal links in index or with <code>\edlineref</code> are now targeted to the top and not longer to the bottom of the lines they refer to. . . . .	1
<code>\ledinnote</code> : <code>\ledinnote</code> takes a first optional argument, which is the label for hyper- links. . . . .	239
v1.22.1.	
General: Fix bug (added on v1.22.0) with <code>\Xinplaceofnumber</code> hook. . . . .	1
<code>\prevpage@num</code> : Correct double symbol when using both <code>\parafootsep</code> and <code>\Xsymlinenum</code> . . . . .	152
v1.23.0.	
<code>\@edtext@level</code> : The boolean <code>\if@edtext@</code> becomes the counter <code>\edtext@level</code> . . . . .	106
General: Add <code>\Xboxlinenumalign</code> and <code>\Xendboxlinenumalign</code> . . . . .	33
Add <code>\Xboxstartlinenum</code> , <code>\Xendboxstartlinenum</code> , <code>\Xboxendlinenum</code> , <code>\Xendboxendlinenum</code> . . . . .	33
Allow use of <code>\sameword</code> with <code>inputenc</code> managing of UTF-8. . . . .	1
Compatibility between <code>nofamiliar/nocriticals</code> option and <code>minipage/ledgroup</code> . . . . .	1
Error message when using <code>\beginnumbering... \endnumbering</code> without <code>\pstart</code> . . . . .	1
Fix bug with <code>\sameword</code> when the lemma overlaps multiple line. . . . .	25
Fix bug with <code>\sameword</code> when the same lemma is used for multiple notes or for nested <code>\edtexts</code> . . . . .	25
Fix bug with <code>\skipnumbering</code> called immediately after a <code>\pstart</code> . . . . .	1
Fix error of <code>\iftrue</code> not closed. . . . .	1
Fix spurious space with <code>\skipnumbering</code> (bug added on v1.21.0). . . . .	1
New tools to ensure the line-list file uses the right version of commands when upgrad- ing the <code>eledmac</code> version. . . . .	1
Optional argument of <code>\sameword</code> can be a comma-separated list of <code>\edtext</code> depth. . . . .	25
<code>\lemma</code> : Fix spurious space after <code>\lemma</code> command . . . . .	110
<code>\newseries@</code> : Prevent spurious spaces when <code>\Afootnote</code> and similar commands are followed by spaces (bug added on 1.0.0). . . . .	198
<code>\sameword</code> : In order to allow use of <code>\sameword</code> with <code>inputenc</code> , we detokenize its manda- tory argument before using it in control sequence names. . . . .	114
v1.23.1.	
General: Fix bug with <code>\lemma</code> command in the right side. . . . .	1
v1.23.2.	
General: Compatibility with $\text{\LaTeX}$ '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. . . . .	234
<code>\xympar</code> : Patching <code>\xympar</code> instead of redefining it . . . . .	226
General: <code>\makecol</code> , <code>\reinserts</code> and <code>\doclearpage</code> are patched instead of begin redefined . . . . .	215
<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> . . . . .	215
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 . . . . .	169
Suppress indexing command specific to <code>memoir</code> . . . . .	240
<code>\endminipage</code> : Patch <code>\endminipage</code> instead of redefining it. . . . .	234
<code>\initnumbering@quote</code> : <code>\initnumbering@sectcmd</code> becomes <code>\initnumbering@quote</code> . . . . .	272
<code>\l@advance@parledgroup@beforenormalnotes</code> : Some conde of <code>\l@dumboxmpfoot</code> moved to <code>\l@advance@parledegroupp@beforenormalnotes</code> . . . . .	234
<code>\newseries@</code> : One endnotes file by series. . . . .	201
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>\Xbhooknote</code> and <code>\bhooknoteX</code> when using them to insert text and not to execute code. . . . .	1
New tools to number stanzas . . . . .	1
v2.1.1.	
General: Fix bug with <code>\ledpbsetting{before}</code> . . . . .	1



v2.1.2.	
General: Fix bug with lineation by pstart and tabular environments (added in 2.1.0).	1
v2.1.3.	
General: <code>\Xhangindent</code> and <code>\hangindentX</code> work now with all the paragraphs in the note.	1
<code>\Xnoindent</code> and <code>\noindentX</code> work now again (broken in 2.0.0).	1
Change some internal code in order to provide compatibility with $\TeX$ 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.	183
v2.2.0.	
General: Fix bug with combination of <code>\onehalfspacing</code> and two columns and three columns notes typeset.	1
Fix bug with some setting command and optimization option.	1
Fix spurious space with paragraphed critical notes when using $\text{Lua}\TeX$ .	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>\Xendbeforesymmlinenumber</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 $\text{ETEX}$ 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>bid</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>\Xhooknote</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> .	30
Option <code>linerangesep</code> for critical footnotes and endnotes.	30
<code>\footnoteoptions@</code> : First argument of <code>\footnoteoption@</code> is now mandatory, not optional.	138