

# reledmac

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

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 **ledmac** package, which was based on **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 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 all cases). Examples starting by “1-” are for basic uses, those starting by “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](https://github.com) to access my page ([maieul/ledmac](https://github.com/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>9</b>
1.1 Aim of the package . . . . .	9
1.2 History . . . . .	10
1.2.1 edmac . . . . .	10
1.2.2 ledmac . . . . .	12
1.2.3 eledmac . . . . .	12

---

<sup>\*</sup>This file (**reledmac.dtx**) has version number v2.1.2, last revised 2015/08/26.

<sup>†</sup>`maieul at maieul dot net`

1.2.4 <code>reledmac</code> . . . . .	12
1.3 List of works edited with (r)(e)ledmac . . . . .	12
<b>2 How the package works</b>	<b>12</b>
<b>3 Options</b>	<b>13</b>
3.1 Specific features . . . . .	13
3.2 Optimize package's performance . . . . .	13
<b>4 Text lines and paragraphs numbering</b>	<b>14</b>
4.1 Text lines numbering . . . . .	14
4.2 Paragraphs . . . . .	15
4.2.1 Basis . . . . .	15
4.2.2 Producing automatically <code>\pstart... \pend</code> . . . . .	15
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 . . . . .	17
4.3.1 Disabling lineation . . . . .	17
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 . . . . .	19
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 . . . . .	23
5.2.6 Changing the names of commands for critical apparatus . . . . .	24
5.3 Disambiguation of identical words in the apparatus . . . . .	24
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 . . . . .	25

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 . . . . .	27
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>28</b>
6.1 Notes arrangement in a series . . . . .	29
6.2 Control line number printing . . . . .	29
6.2.1 Print line number only at first time . . . . .	29
6.2.2 Abbreviate line range . . . . .	30
6.2.3 Disable line number . . . . .	31
6.2.4 Printing pstart number . . . . .	31
6.2.5 Printing stanza number . . . . .	31
6.2.6 Space around number . . . . .	31
6.2.7 Space around line symbol . . . . .	32
6.2.8 Space in place of number . . . . .	32
6.2.9 Boxing line number and line symbol . . . . .	32
6.3 Separator between the lemma and the note . . . . .	33
6.3.1 For footnotes . . . . .	33
6.3.2 For endnotes . . . . .	33
6.4 Font style . . . . .	34
6.4.1 For line number . . . . .	34
6.4.2 For the lemma . . . . .	34
6.5 Styles of notes content . . . . .	34
6.6 Arbitrary code at the beginning of notes . . . . .	35
6.7 Options for footnotes in columns . . . . .	35
6.7.1 Alignment . . . . .	35
6.7.2 Size of the columns . . . . .	35
6.8 Options for paragraphed footnotes . . . . .	36
6.8.1 Mark separation of notes . . . . .	36
6.8.2 Ragging . . . . .	36
6.9 Options for block of notes . . . . .	36
6.9.1 Text before notes . . . . .	36
6.9.2 Spacing . . . . .	36
6.9.3 Rule . . . . .	37
6.9.4 Maximum height . . . . .	37
6.10 Footnotes and the <code>reledpar</code> columns . . . . .	37
6.11 Endnotes in one paragraph . . . . .	37
<b>7 Fonts</b> . . . . .	<b>37</b>

<b>8 Verse</b>	<b>39</b>
8.1 Basic . . . . .	39
8.2 Define stanza indents . . . . .	39
8.3 Repeating stanza indents . . . . .	39
8.4 Manual stanza indent . . . . .	40
8.5 Stanza breaking . . . . .	40
8.6 Hanging symbol . . . . .	40
8.7 Long verse and page break . . . . .	41
8.8 Content before/after verses . . . . .	41
8.9 Numbering stanza . . . . .	41
8.10 Various tools . . . . .	42
<b>9 Grouping</b>	<b>42</b>
<b>10 Cross referencing</b>	<b>42</b>
10.1 Basic use . . . . .	43
10.2 Refer to a critical notes . . . . .	43
10.3 Cross-referencing which return a number in any case . . . . .	43
10.3.1 Cross-referencing in order to define line number of a critical note . . . . .	44
10.4 Not automatic cross-referencing . . . . .	44
10.5 Normal L <sup>A</sup> T <sub>E</sub> X cross-referencing . . . . .	44
10.6 References to lines commented in the apparatus . . . . .	44
<b>11 Side notes</b>	<b>45</b>
11.1 Basis . . . . .	45
11.2 Setting . . . . .	46
11.2.1 Width . . . . .	46
11.2.2 Vertical position . . . . .	46
11.2.3 Distance to the main text . . . . .	46
11.2.4 Separator between notes . . . . .	46
<b>12 Indexing</b>	<b>46</b>
12.1 Basis . . . . .	46
12.2 Separator between page and line numbers . . . . .	47
12.3 Using xindy . . . . .	47
12.4 Advanced setting . . . . .	48
<b>13 Tabular material</b>	<b>48</b>
<b>14 Sectioning commands</b>	<b>51</b>
14.1 Sectioning commands without line numbers or critical notes . . . . .	51
14.2 Sectioning commands with line numbering and critical notes . . . . .	52
14.3 Optimization . . . . .	53
<b>15 Quotation environments</b>	<b>53</b>

<b>16 Page breaks</b>	<b>53</b>
16.1 Control page breaking . . . . .	53
16.2 Prevent page break in a long verses . . . . .	53
<b>17 Miscellaneous</b>	<b>54</b>
17.1 Known and suspected limitations . . . . .	54
17.2 ‘No room for a new’ . . . . .	54
17.3 Marginal notes . . . . .	55
17.4 Paragraph shape . . . . .	55
17.5 Paragraphed footnotes . . . . .	55
17.6 Use with other packages . . . . .	55
17.7 Parallel typesetting . . . . .	57
<b>I Implementation overview</b>	<b>58</b>
<b>II Preliminaries</b>	<b>58</b>
II.1 Links with original edmac . . . . .	58
II.2 Package declaration . . . . .	58
II.3 Package options . . . . .	59
II.4 Loading packages . . . . .	60
II.5 Boolean flags . . . . .	61
II.6 Messages . . . . .	61
II.7 Gobbling . . . . .	67
II.8 Miscellaneous commands . . . . .	67
II.9 Prepare reledpar . . . . .	67
<b>III Sectioning commands</b>	<b>68</b>
<b>IV List macros</b>	<b>72</b>
<b>V Line counting</b>	<b>73</b>
V.1 Choosing the system of lineation . . . . .	73
V.2 Line number margin . . . . .	75
V.3 Line number initialization and increment . . . . .	76
V.4 Line number locking . . . . .	77
V.5 Line number style . . . . .	78
V.6 Line number printing . . . . .	78
V.7 Line number counters and lists . . . . .	79
V.8 Line number locking counter . . . . .	80
V.9 Line number associated to lemma . . . . .	81
V.10 Reading the line-list file . . . . .	84
V.11 Commands within the line-list file . . . . .	86
V.12 Writing to the line-list file . . . . .	97

<b>VI Marking text for notes</b>	<b>102</b>
VI.1 <code>\edtext</code> itself . . . . .	103
VI.2 Substitute lemma . . . . .	109
VI.3 Substitute line numbers . . . . .	110
VI.4 Lemma disambiguation . . . . .	111
<b>VII Paragraph decomposition and reassembly</b>	<b>117</b>
VII.1 Boxes, counters, <code>\pstart</code> and <code>\pend</code> . . . . .	117
VII.2 Processing one line . . . . .	122
VII.2.1 General process . . . . .	122
VII.2.2 Process for “normal” line . . . . .	123
VII.2.3 Process for line containing <code>\eledsection</code> command . . . . .	124
VII.2.4 Hooks . . . . .	124
VII.2.5 Sidenotes and marginal line number initialization . . . . .	125
<b>VIII Line and page number computation</b>	<b>125</b>
<b>IX Line number printing</b>	<b>128</b>
<b>X Pstart number printing in side</b>	<b>132</b>
<b>XI Restoring footnotes and penalties</b>	<b>134</b>
XI.1 Add insertions to the vertical list . . . . .	134
XI.2 Penalties . . . . .	135
XI.3 Printing leftover notes . . . . .	136
<b>XII Critical footnotes</b>	<b>137</b>
XII.1 Fonts . . . . .	137
XII.2 Individual note options . . . . .	137
XII.3 Notes language . . . . .	138
XII.4 General survey of the way we manage notes . . . . .	139
XII.5 General setup . . . . .	139
XII.6 Footnotes arrangement . . . . .	140
XII.6.1 User level macro . . . . .	140
XII.6.2 Normal footnote . . . . .	140
XII.6.3 Paragraphed footnotes . . . . .	144
XII.6.4 Columnar footnotes . . . . .	151
XII.7 Critical notes presentation . . . . .	157
XII.7.1 Font tools . . . . .	157
XII.7.2 Pstart number in footnote . . . . .	158
XII.7.3 Line number printing . . . . .	158
<b>XIII Familiar footnotes</b>	<b>166</b>
XIII.1 Adjacent footnotes . . . . .	166
XIII.2 Regular footnotes for numbered texts . . . . .	167
XIII.3 Footnote formats . . . . .	168
XIII.4 Footnote arrangement . . . . .	169

XIII.4.1 User level macro . . . . .	169
XIII.4.2 Normal footnotes . . . . .	169
XIII.4.3 Two columns footnotes . . . . .	174
XIII.4.4 Three columns footnotes . . . . .	176
XIII.4.5 Paragraphed footnotes . . . . .	178
<b>XIV Code common to both critical and familiar footnote in normal arrangement</b>	<b>181</b>
<b>XV Footnotes' width for two columns</b>	<b>182</b>
<b>XVI Footnotes' order</b>	<b>183</b>
<b>XVII Footnotes' rule</b>	<b>184</b>
<b>XVIII Specific skip for first series of footnotes</b>	<b>184</b>
XVIII.0.1 Overview . . . . .	184
XVIII.0.2 User level command . . . . .	185
XVIII.0.3 Internal commands . . . . .	185
<b>XIX Endnotes</b>	<b>186</b>
<b>XX Generate series of notes</b>	<b>193</b>
XX.1 Test if series is still existing . . . . .	193
XX.2 Init specific to <code>reledpar</code> . . . . .	193
XX.3 For critical footnotes . . . . .	194
XX.3.1 Options . . . . .	194
XX.3.2 Create inserts, needed to add notes in foot . . . . .	195
XX.3.3 Create commands for critical apparatus, <code>\Afootnote</code> , <code>\Bfootnote</code> etc. . . . .	195
XX.3.4 Set standard display . . . . .	196
XX.4 For familiar footnotes . . . . .	197
XX.4.1 Options . . . . .	197
XX.4.2 Create tools for familiar footnotes ( <code>\footnotex</code> ) . . . . .	197
XX.5 The endnotes . . . . .	198
XX.5.1 The auxiliary file . . . . .	198
XX.5.2 The main macro . . . . .	198
XX.5.3 The options . . . . .	199
XX.6 Init standards series (A,B,C,D,E) . . . . .	200
<b>XXI Setting series display</b>	<b>200</b>
XXI.1 Change series order . . . . .	200
XXI.2 Test series order . . . . .	201
XXI.2.1 Get the first series . . . . .	201
XXI.3 Series setting . . . . .	201
XXI.3.1 General way of working . . . . .	201
XXI.3.2 Tools to set options . . . . .	202

XXI.3.3 Tools to generate options commands . . . . .	203
XXI.3.4 Options for critical notes . . . . .	205
XXI.3.5 Options for familiar notes . . . . .	206
XXI.3.6 Options for endnotes . . . . .	206
XXI.4 Hooks for a particular footnote . . . . .	207
XXI.5 Alias . . . . .	207
<b>XXII Output routine</b>	<b>208</b>
XXII.0.1 Page number management . . . . .	208
XXII.0.2 Extra footnotes output . . . . .	208
XXII.0.3 Standard output's commands patching . . . . .	210
<b>XXIII Cross referencing</b>	<b>213</b>
<b>XXIV Side notes</b>	<b>222</b>
<b>XXV Minipages and such</b>	<b>228</b>
<b>XXVI Indexing</b>	<b>232</b>
<b>XXVII Verse</b>	<b>240</b>
XXVII.1 Hanging symbol management . . . . .	240
XXVII.2 Using & character . . . . .	240
XXVII.3 Code category setting . . . . .	241
XXVII.4 Stanza count and indent . . . . .	241
XXVII.5 Numbering stanza . . . . .	242
XXVII.6 Stanza number in note . . . . .	243
XXVII.7 Main work . . . . .	244
XXVII.8 Restore catcode and penalties . . . . .	246
<b>XXVIII Arrays and tables</b>	<b>246</b>
XXVIII.1 Preamble: macro as environment . . . . .	246
XXVIII.2 Tabular environments . . . . .	249
XXVIII.2.1 Disabling and restoring commands . . . . .	250
XXVIII.2.2 Counters, boxes and lengths . . . . .	253
XXVIII.2.3 Tabular typesetting . . . . .	257
XXVIII.2.4 Environments . . . . .	268
<b>XXIX Quotation's commands</b>	<b>268</b>
<b>XXX Section's title commands</b>	<b>269</b>
XXX.1 Commands to disable some feature . . . . .	269
XXX.2 General overview . . . . .	270
XXX.3 \beforeeledchapter command . . . . .	270
XXX.4 Auxiliary commands . . . . .	271
XXX.5 Patching standard commands . . . . .	272
XXX.6 Main code of \eledxxx commands . . . . .	276



XXX.7 Macros written in the auxiliary file . . . . .	279
<b>XXXI Page breaking or no page breaking depending of specific lines</b>	<b>281</b>
<b>XXXII Long verse: prevents being separated by a page break</b>	<b>283</b>
<b>XXXIII Compatibility with eledmac</b>	<b>283</b>
<b>Appendix A Some things to do when changing version</b>	<b>286</b>
Appendix A.1 Migrating from edmac to ledmac . . . . .	286
Appendix A.2 Migration from ledmac to eledmac . . . . .	287
Appendix A.3 Migration to eledmac 1.5.1 . . . . .	288
Appendix A.4 Migration to eledmac 1.12.0 . . . . .	288
Appendix A.5 Migration to eledmac 17.1 . . . . .	289
Appendix A.6 Migration to eledmac 1.21.0 . . . . .	289
Appendix A.6.1 <code>\Xledsetnormalparstuff</code> and <code>\ledsetnormalparstuff</code>	289
Appendix A.6.2 Endnotes . . . . .	289
Appendix A.7 Migration to eledmac 1.22.0 . . . . .	289
Appendix A.8 Migration to eledmac 1.23.0 . . . . .	289
Appendix A.9 Migration from eledmac to reledmac . . . . .	290
Appendix A.9.1 Risk of ‘no room for a new’ . . . . .	290
Appendix A.9.2 Multiple indices with memoir . . . . .	290
Appendix A.9.3 Deprecated commands and options . . . . .	290
Appendix A.9.4 <code>\renewcommandreplaced by command</code> . . . . .	291
Appendix A.9.5 Commands the names of which have been changed . . . . .	291
Appendix A.9.6 Endnotes . . . . .	293
Appendix A.9.7 Z Series . . . . .	293
Appendix A.9.8 Internal commands . . . . .	293
Appendix A.10 Migration to reledmac 2.1.0 . . . . .	293
<b>References</b>	<b>294</b>
<b>Index</b>	<b>294</b>
<b>Change History</b>	<b>333</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 for both prose and verse;
- multiple series of the 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 some other  $\text{\LaTeX}$  packages for critical edition typesetting. However, the aim of `reledmac` is to provide an “all in one” and flexible tool in the field of critical edition.

Any suggestion for new features are welcome.

This manual contains a general description of how to use `reledmac` and the complete source code for the package, with extensive documentation (in sections I and following, numbered in Roman numeric). It ends with a list of actions to do when migrating from some version to other, a change history and an index to the source code.

We do not suggest that you 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 the earlier sections.

But no documentation, however thorough, can cover every question that comes up, and many can be answered quickly by consultation of the code. On a first reading, we suggest that you should 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 T<sub>E</sub>X 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 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 T<sub>E</sub>X 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.

At the time of writing (1994), we are pleased to be able to say that edmac is being used for 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 *Rithmachia* of Werinher von Tegernsee,<sup>7</sup> a middle-Dutch romance epic on the Crusades,<sup>8</sup> a seventeenth-century Hungarian politico-philosophical tract,<sup>9</sup> an anonymous Latin compilation from Hungary entitled *Sermones Compilati in Studio Generali Quinqueecclesiensi in Regno Ungarie*,<sup>10</sup> the collected letters and papers of Leibniz,<sup>11</sup> Theodosius’s *Spherics*, the German *Algorismus* of Sacrobosco, the Sanskrit text of the *Kāśikāvṛtti* of Vāmana and Jayāditya,<sup>12</sup> and the English texts of Thomas Middleton’s collected works.

<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 *Rithmachia* des Werinher von Tegernsee’, *ibid.*

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

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

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

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

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

### 1.2.2 `ledmac`

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

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

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

This port was called `ledmac` (L<sup>A</sup>T<sub>E</sub>X `edmac`).

Since July 2011, `ledmac` is maintained by Maïeul Rouquette. Is it more and more powerful and flexible, but also more and more divergent from original TeX macro.

### 1.2.3 `eledmac`

Important changes were put in version 1.0, to make `ledmac` more easily extensible (see 6 p. 28). 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. 287.

### 1.2.4 `reledmac`

`eledmac` has facilitated the creation of customized critical edition. However, this was made in a non systematic way in method to customize them.

In other side, many deprecated commands were kept, and many technical debt was accumulated, blocking the futur evolution.

For all these reasons, Maïeul Rouquette decided to make a spring cleaning. As some commands name were changed, the ascendant compatibility was (a little) broken.

A new name was selected: `reledmac` (extended renewed `eledmac`).

To migrate from `eledmac` to `reledmac`, please read Appendix A.9 p. 290.

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

A collaborative list of works edited with (r)(e)ledmac is available on [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  $\text{\LaTeX}$  itself. Although your textual apparatus and line numbers will be printed even on the first run, it takes two more passes through  $\text{\LaTeX}$  to be sure that everything gets to its right place. Any changes you make to the input file may similarly require three passes to get everything to the right

place, if the changes alter the number of lines or notes. `reledmac` will tell you that you need to make more runs, when it notices, 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 `ℒTEX` 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: options which provide specific features and option which optimize package's performance. It is advised to read the relevant parts of the handbook before reading the first type of option, but you can look at the second type in you first reading of handbook.

### 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. 291).

**`nopbinverse`** prevents page break inside verses.

**`noquotation`** by default, the quotation environment is redefined inside numbered text. You can disable this redefinition with `noquotation` (see 15 p. 53).

**`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. 47).

**`widthliketwocolumns`** set the width of the text disposed on one column to be the same as the width of the text disposed on two parallel columns with `reledpar`. This is useful when alternating between normal and parallel typesetting.

### 3.2 Optimize package's 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. 52).

**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 to use of series of critical notes and new series of normal notes inside `minipage` and `ledgroup` environments (see 9 p. 42). 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 all of the 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` `\endnumbering` Each section of numbered text must be preceded by `\beginnumbering` and followed by `\endnumbering`, like this:

```
\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 Basis

`\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 Producing automatically `\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` by 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. 39) or `reledpar` (see 17.7 p. 57).

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

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

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

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

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

On each `\beginnumbering` the numbering restarts.

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

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

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



### 4.2.6 Languages written in Right to Left

If you use languages written in 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 problem described below should not appear with 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 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 say

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

and say `\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` `\numberlinetrue`.

### 4.3.2 Setting lineation start and step

`\firstlinenum`  
`\linenumincrement`

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

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

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

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

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

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

to have numbers printed on prime-numbered lines only. There must be no spaces within the definition which consists of comma-separated integer numbers. The numbers can be in any order but it is easier to read if you put them in numerical order. Either omitting the definition of `\linenumberlist` or following the vacuous 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 value for `<location>` is one out of the list `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 that in effect at the `\pend` of a numbered paragraph. Apart from an initial setting for `\linenummargin`, only change it 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 the current paragraph).

### 4.3.5 Other settings

`\leftlinenum` When a marginal line number is to be printed, there are a lot of ways to display it.  
`\rightlinenum` You can redefine `\leftlinenum` and `\rightlinenum` to change the way marginal line  
`\linenumsep` 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 the line numbering starts at 1 for the first line of a section and steps up by one for each line thereafter. There are various common modifications of this system, however; the commands described here allow you to put such modifications into effect.

### 4.4.1 Sublineation

`\startsub` You insert the `\startsub` and `\endsub` commands in your text to turn sub-lineation  
`\endsub` on and off. In plays, for example, stage directions 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 that changes in the middle.

### 4.4.2 Locking lineation

`\startlock` The `\startlock` command, used in running text, locks the line number at its current  
`\endlock` value, until you say `\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. 39.

`\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.  
(This assumes that, on the basis of the settings of the previous parameters, it is nec-  
essary to display a line number for this line.) You specify your preference using  
`\lockdisp{<arg>}`; its argument is a word, either `first`, `last`, or `all`. The package  
initially sets this as `\lockdisp{first}`.

### 4.4.3 Setting and changing line number

`\setline` In some cases you may want to modify the line numbers that are automatically cal-  
`\advanceline` culated: if you are printing only fragments of a work but want to print line num-  
bers 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. `\advance` takes one argument, an amount that should be added to the current line number; it may be positive or negative.

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

#### 4.4.4 Line number style

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

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

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

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

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

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

Note that with the **Alph** or **alph** styles, ‘numbers’ must be between 1 and 26 inclusive.

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

#### 4.4.5 Skipping and hiding number

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

`\hidenumbering` When inserted into a numbered line the macro `\hidenumbering` causes the number for that particular line to be hidden; namely, no line number will print. Note that if you use it in `\stanza`, you must call it at the beginning of the verse.

#### 4.4.6 Execute code at each line

`\dolinehook` `\doinsidelinehook` `reledmac` provides to advanced feature for user. 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 case, the latter is more useful than the first. The file `examples/2-line_numbers_in_header.tex` provides an example of use in order to print the first and last line number of a page in the header.

## 5 Apparatus commands

### 5.1 Terminology

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

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

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

So :

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

### 5.2 Critical notes

#### 5.2.1 The lemma

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

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

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

For example:

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

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

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

I am happy : <code>\edtext{I saw my friend</code>	1	I am happy : I saw my friend Smith on
<code>\edtext{Smith}{\Afootnote{Jones</code>	2	Tuesday.
<code>C, D.}}</code> on Tuesday.]{		
<code>\Bfootnote{The date was</code>		
July 16, 1954.}		1 Smith] Jones C, D.
}		
		1-2 I saw my friend Smith on Tuesday.] The
		date was July 16, 1954.

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

### 5.2.2 Footnotes

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

`\Afootnote` Five separate series of the footnotes are maintained; each macro takes one argument like `\Afootnote{ $\langle text \rangle$ }`. 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. 6.2.2 p. 30).
- `nonum` to disable line numbering for this note.
- `nosep` to disable the lemma separator for this note.

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

### 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[ $\langle option \rangle$ ]{ $\langle text \rangle$ }`.

$\langle option \rangle$  can contain a comma separated list of values. Allowed values are:

- `fulllines` to disable `\Xendtwolines` and `\Xendmorethantwolines` features for this particular note (cf. 6.2.2 p. 30).
- `nonum` to disable line number for this particular note.
- `nosep` to disable the lemma separator for this particular note.

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

`\doendnotesbysection` However, you may want to print the endnotes of one given series covering the first

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

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

Note that by default inside endnotes no separator is used between the lemma and the content. However you can use the `\Xendlemmaseparator` macro to define one (6.3.2 p. 33).

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}
```

#### 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`, 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. 42) 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 say something you find more meaningful, like `\variant`.

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

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

## 5.3 Disambiguation of identical words in the apparatus

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

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



### 5.3.1 Basic use

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

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

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

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

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

For instance, in Greek, “ $\alpha$ ” has two possible unicode numbers:

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

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

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

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

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

---

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

```

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

```

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

Consequently, you have to tell to reledmac which instance of `\sameword` in the first argument of `\edtext` you want to reference:

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

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

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

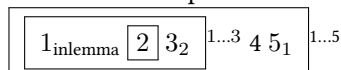
```

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

```

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

In the following example figure, each framed box represents an `\edtext` level. Each number is an occurrence of `\sameword`. After a framed box, the text in superscript represents the content of `\lemma` for that `\edtext` level. The text in subscript at the right of a number represents the content of the optional argument of `\sameword`.



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

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

### 5.3.4 Customizing

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

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

## 5.4 Familiar notes

### 5.4.1 Basic use

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

### 5.4.2 Customizing mark

`\thefootnoteA` Each series uses a set of macros for styling the marks. The mark numbering scheme of  
`\bodyfootmarkA` series A is defined by the `\thefootnoteA` macro; the default is:  
`\footfootmarkA` `\renewcommand*{\thefootnoteA}{\arabic{footnoteA}}`  
 The appearance of the mark in the text is controlled by `\bodyfootmarkA` which is defined as:  
`\newcommand*{\bodyfootmarkA}{%`  
`\hbox{\textsuperscript{\normalfont\@nameuse{@thefnmarkA}}}`  
 The command `\footfootmarkA` controls the appearance of the mark at the start of the footnote text. It is defined as:  
`\newcommand*{\footfootmarkA}{\textsuperscript{\@nameuse{@thefnmarkA}}}`  
 There are similar command triples for the other series.

### 5.4.3 Separator for multiple footnotes

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

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

## 5.5 Changing series

### 5.5.1 Create a new series

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

### 5.5.2 Delete series

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

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

### 5.5.3 Series order

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

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` 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 concern all series.

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>16</sup>, it will be relative to fontsize of the footnote, except for commands concerning

---

<sup>16</sup>Like `em` which is the width of a `mg`.

the place kept by the notes — including blank space.

There is also name convention:

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

## 6.1 Notes arrangement in a series

`\Xarrangement`  
`\arrangementX`

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

Use `\Xarrangement[⟨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;
- `twocol` formats them as separate paragraphs, but in two columns;
- `threecol`, in three columns.
- `normal`, restore normal arrangement.

You should set up the page layout parameters, and in particular the `\baselineskip` of the footnotes, before you call this macro because its action depends on these; too much or too little space will be allotted for the notes on the page if these macros use the wrong values.<sup>17</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 again note arrangement.

`\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⟩]`.

<sup>17</sup>There is one tiny proviso about using paragraphed notes: you should not force any explicit line-breaks inside such notes: do not use `\par`, `\break`, or `\penalty=-10000`. If you must have a line-break for some obscure reason, just suggest the break very strongly: `\penalty=-9999` will do the trick. XII.6.3 p. 147 explains why this restriction is necessary.

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⟩]`, the 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). 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 not robust command.

`\Xsymlinenum`

### 6.2.2 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`  
`\Xendmorethantwolines`  
`\Xendtwolinesbutnotmore`

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

### 6.2.3 Disable line number

`\Xnonumber` 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.

### 6.2.4 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).

### 6.2.5 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.

### 6.2.6 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.

`\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).

### 6.2.7 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`.

### 6.2.8 Space in place of number

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

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

### 6.2.9 Boxing line number and line symbol

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

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

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

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

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

**R** to align right;

**C** to center.

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

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

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



12-23  
24ff.

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

### 6.3 Separator between the lemma and the note

#### 6.3.1 For footnotes

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

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

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

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

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

#### 6.3.2 For endnotes

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

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

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

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

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

## 6.4 Font style

### 6.4.1 For line number

<code>\Xnotenumfont</code>	<code>\Xnotenumfont[⟨s⟩]{⟨command⟩}</code> is used to change the font style for line numbers in critical footnotes ; <code>⟨command⟩</code> must be one (or more) switching command, like <code>\bfseries</code> .
<code>\Xendnotenumfont</code>	<code>\Xendnotenumfont[⟨s⟩]{⟨command⟩}</code> is used to change the font style for line numbers in critical footnotes. <code>⟨command⟩</code> must be one (or more) switching command, like <code>\bfseries</code> .
<code>\notenumfontX</code>	<code>\notenumfontX[⟨s⟩]{⟨command⟩}</code> is used to change the font style for note numbers in familiar footnotes. <code>⟨command⟩</code> must be one (or more) switching command, like <code>\bfseries</code> .
<code>\Xnotefontsize</code>	<code>\Xnotefontsize[⟨s⟩]{⟨command⟩}</code> is used to define the font size of critical footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .
<code>\notefontsizeX</code>	<code>\notefontsizeX[⟨s⟩]{⟨command⟩}</code> is used to define the font size of familiar footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .
<code>\Xendnotefontsize</code>	<code>\Xendnotefontsize[⟨s⟩]{⟨l⟩}</code> is used to define the font size of end critical footnotes of the series. The default value is <code>\footnotesize</code> . The <code>⟨command⟩</code> must not be a size in pt, but a standard $\TeX$ size, like <code>\small</code> .

### 6.4.2 For the lemma

<code>\Xlemmadisablefontselection</code>	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 <code>\Xlemmadisablefontselection[⟨s⟩]</code> command allows to disable it for a specific series.
<code>\Xendlemmadisablefontselection</code>	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 <code>\Xendlemmadisablefontselection[⟨s⟩]</code> to disable it for a specific series.

## 6.5 Styles of notes content

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

## 6.6 Arbitrary code at the beginning of notes

The three next commands add an 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.
---	--

## 6.7 Options for footnotes in columns

### 6.7.1 Alignment

`\Xcolalign` By default, texts in footnotes in two or three columns are flushed left 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`).

### 6.7.2 Size of the columns

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

<code>\Xhsizetwocol</code> <code>\Xhsizethreecol</code> <code>\hsizetwocolX</code> <code>\hsizethreecolX</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[⟨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>\hsizetwocol[⟨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[⟨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> .
--	---

## 6.8 Options for paragraphed footnotes

### 6.8.1 Mark separation of notes

`\Xafternote` You can add some space after a note by using `\Xafternote[⟨s⟩]{⟨l⟩}` (for critical footnotes) or `\afternoteX[⟨s⟩]{⟨l⟩}` (for familiar footnotes). The default value is 1em plus .4em minus .4em.

`\Xparafootsep` For paragraphed footnotes (see below), you can choose the separator between each note by using `\Xparafootsep[⟨s⟩]{⟨text⟩}` for critical notes and `\parafootsepX` for familiar notes. A common separator is the double pipe (`||`), which you can set by using `\parafootsep{$\parallel$}`.

`\parafootsepX` 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.

### 6.8.2 Ragging

`\Xragged` Text in paragraphed critical notes is justified, but you can use `\Xragged[⟨s⟩]{L}` if you want it to be ragged left, or `\Xragged[⟨s⟩]{R}` if you want it to be ragged right.

`\raggedX` Text in paragraphed footnotes is justified, but you can use `\raggedX[⟨s⟩]{L}` if you want it to be ragged left, or `\raggedX[⟨s⟩]{R}` if you want it to be ragged right.

## 6.9 Options for block of notes

### 6.9.1 Text before notes

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

### 6.9.2 Spacing

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

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

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

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

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

`\prenotesX` You can want the space before the first printed (in a page) series of familiar notes not to be the same as before other series. Default value is 0pt. You can do it with `\prenotesX{⟨l⟩}`. You can disable this feature by setting the length to 0pt.

### 6.9.3 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  $\TeX$  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  $\TeX$  footnote rule, which is used by `reledmac`, adds 2.6pt. These 2.6pt are not changed by this command.**

### 6.9.4 Maximum height

`\Xmaxhnotes` By default, one series of critical notes can take 80% of the page size, 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.

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

## 6.10 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.

## 6.11 Endnotes in one paragraph

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

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

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

## 7 Fonts

One of the most important features of the appearance of the notes, and indeed of your whole document, will be the fonts used. We will first describe the commands that give

you control over the use of fonts in the different structural elements of the document, especially within the notes, and then in subsequent sections specify how these commands are used.

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

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

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

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

`\endashchar` A relatively trivial matter relates to punctuation. In your footnotes, there will sometimes be spans of line numbers like this: 12–34, or 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 the en-dash or full stop in the same places as a text font. If you (or your macros) just typed `$_oldstyle 12--34$` or `$_oldstyle 55.6$` you would get ‘12”34’ and ‘55▷6’. So we define `\endashchar` and `\fullstop`, which produce an en-dash and a full stop respectively from the normal document font, whatever font you are using for the numbers. These two macros are used in the macros which format the line numbers in the margins and footnotes, instead of explicit punctuation. We also define an `\rbracket` macro for the right square bracket printed at the end of the lemma in many styles of textual notes (including `reledmac`’s standard style). For `polyglossia`, when the lemma is RTL, the bracket automatically switches to a left bracket.

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

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

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

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

## 8 Verse

### 8.1 Basic

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

### 8.2 Define stanza indents

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

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

To specify these multiples one invokes, for example  
`\setstanzaindents{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 `\sethangingsymbol:` see p. 8.6 p. 41.

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 say they are repeated, defining the value of the `stanzaindentsrepetition` 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. 288.**

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{stanzaindentrepetition}{0}
```

The macros make no restriction on the number of lines in a stanza. Stanza indentation values (and penalty values) obey T<sub>E</sub>X's grouping conventions, so if one stanza among several has a different structure, its indentations (penalties) may be set within a group; the prior values will be restored when the group ends.

## 8.4 Manual stanza indent

`\stanzaindent`  
`\stanzaindent*`

You can set the indent of some specific verse by calling `\stanzaindent{⟨value⟩}` at the beginning of the verse, before any other character. In this case, the indent defined by `\setstanzaindent` 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 T<sub>E</sub>X, 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. 53 for further details.

## 8.8 Content before/after verses

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

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

## 8.9 Numbering stanza

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

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

You can use this feature in combination with `\Xstanza` (6.2.5 p. 31).

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

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

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

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

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

### 8.10 Various tools

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

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

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

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

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

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

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

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

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

`\edlineref`

`\sublineref` 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.

`\pstartref`

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

You will be warned if you say `\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
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` Where #1 stands for the reference.

`\xlineref` However, there are situations in which you will want `reledmac` to return a number without displaying any warning messages about undefined labels or the like: if you want to use the reference in a context where `ℒTeX` is looking for a number, such a warning will lead to a complaint that the number is missing. This is the case for references used within the argument to `\linenum`, for example (see 5.2.5 p. 23).

`\xsublineref`

`\xpstartref`

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

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

- 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. 23 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 say ‘`\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 $\text{\LaTeX}$ 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. 23) to refer to these labels, `reledmac` will use your line settings to refer to the passage.

`\appref`

`\apprefwithpage`

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

`\setapprefprefixsingle`

`\setapprefprefixmore`

If you use `\setapprefprefixsingle{<prefix>}`, `<prefix>` will be printed before the 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`

`\Xmorethantwolinesappref`

`\Xtwolinesbutnotmoreappref`

`\Xtwolinesonlyinsamepage`

If you use `\Xtwolines`, `\Xmorethantwolines`, `\Xtwolinesbutnotmore` and/or `\Xtwolinesonlyinsamepage` (6.2.2 p. 30) *without the optional series argument*, the setting will also be available for `\appref`.

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`

`\Xendmorethantwolinesapprefwithpage`

`\Xendtwolinesbutnotmoreapprefwithpage`

`\Xendtwolinesonlyinsamepageapprefwithpage`

If you use one of `\Xendtwolines`, `\Xendmorethantwolines`, `\Xendtwolinesbutnotmore`, `\Xendtwolinesonlyinsamepage` (6.2.2 p. 30) *without the optional series argument*, the setting will also be available for `\apprefwithpage`.

The commands `\Xendtwolinesappref{<text>}`, `\Xendmorethantwolinesappref{<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 Basis

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

`\ledinnernote`

`\ledouternote`

`\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`

`\ledrightnote`

`\ledsidenote`

`\sidenotemargin`

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

The package's default setting is

`\sidenotemargin{right}`

to typeset `\ledsidenotes` in the right hand margin. This is the opposite to 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 Basis

`\edindex` L<sup>A</sup>T<sub>E</sub>X 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 & linenum 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 three times (Xe/Lua) $\LaTeX$ , then run `makeindex` and finally run again (Xe/Lua) $\LaTeX$  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/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>19</sup>.

```
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>20</sup>

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

In any case, you must load `reledmac` with the `xindy` option, in order to generate a `.xdy` file which is specific to your document. This file is needed by the `.xdy` example file

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

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

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>21</sup>.

## 12.4 Advanced setting

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

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

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

## 13 Tabular material

$\TeX$ ’s normal tabular and array environments cannot be used where line numbering is being done; more precisely, they can be used but with odd results, so don’t use them. However, `reledmac` provides some simple tabulation environments that can be line numbered. The environments can also be used in normal unnumbered text.

`edarrayl` 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.

`edarrayc`

`edarrayr`

`edtabularl`

`edtabularc`

`edtabularr`

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

1	2	3
a	bb	ccc
AAA	BB	C

Entries in the environments are the same as for the normal array and tabular environments but there must be no ending `\\` at the end of the last row. *There must be*

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



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

<pre> \begin{edarray} 1 &amp; 2 &amp; 3 &amp; 4 \\ &amp; \spreadmath{F+G+C} &amp; &amp; \\ a &amp; bb &amp; ccc &amp; dddd \end{edarray} </pre>	$ \begin{array}{cccc} 1 & 2 & 3 & 4 \\ & F+G+C & & \\ a & bb & ccc & dddd \end{array} $
---	---

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

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

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

```
\begin{edtabularrr}
1          & 2      & 3      & 4      & 5 \\
Q          & & fd & h      & qwertziohg \\\end{edtabularrr}
```

```

v                                & wptz & x & y & vb \\
g                                & nnn & \edrowfill{3}{5}{\upbracefill} & & \\
\edrowfill{1}{3}{\downbracefill} & & & pq & dgh \\
k                                & & 1 & co & ghweropjklmbvxcys \\
1                                & 2 & 3 & \edrowfill{4}{5}{\hrulefill} & \\
\end{tabularr}

```

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

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

```

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

```

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

```

\begin{edarrayc}
1 & 2 & & & 3 & 4 & \\
a & \edrowfill{2}{3}{\upbracketfill} & & & d & \\
A & B & & & C & D & \\
\end{edarrayc}

```

1	2	3	4
<i>a</i>	$\underbrace{\hspace{1em}}$		<i>d</i>
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>

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

```

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

```

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

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

For example:

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

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

`\edvertline`     The macro `\edvertline{⟨height⟩}` draws a vertical line `⟨height⟩` 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 & & \\\
m & n & o & p & & \\\
k & & L & cvb & \edvertline{4pc} & \\
\end{edarrayr}
```

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

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

## 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 can not add critical notes inside.

## 14.2 Sectioning commands with line numbering and critical notes

You have to use the following commands:

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

Which 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
```

At the first run, you will see only the text. It is normal. At the second run, you will see the formatting. And consequently, at the third run, you will see the table of contents.

For technical reasons, the page break before `\elechapter` can not 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` file, keep memory and make `reledmac` faster.

## 15 Quotation environments

The `quotation` and `quote` environment can be used so that 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 `\start-\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, the 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, the l. 444 will be at the p. *n* and the l. 445 will be at the 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 run 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, not more. It works at the third run, or at fourth run with `reledpar`. By default, when a long verse runs normally 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

<code>\extensionchars</code>	When the package assembles the name of the auxiliary file for a section, it prefixes <code>\extensionchars</code> 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 <code>\renewcommand{\extensionchars}{!}</code> , then you would get temporary files called <code>jobname. !1</code> , <code>jobname. !2</code> , etc.
<code>\ifledfinal</code>	The package can take options. The option ‘final’, which is the default is for final typesetting; this sets <code>\ifledfinal</code> to TRUE. The other option, ‘draft’, may be useful during earlier stages and sets <code>\ifledfinal</code> to FALSE.
<code>\showlemma</code>	<p>The lemma within the text is printed via <code>\showlemma{lemma}</code>. Normally, or with the ‘final’ option, the definition of <code>\showlemma</code> is:</p> <pre>\newcommand*\showlemma[1]{#1}</pre> <p>so it just produces its argument. With the ‘draft’ option it is defined as</p> <pre>\newcommand*\showlemma[1]{\textit{#1}}</pre> <p>so that its argument is typeset in an italic font, which may make it easier to check that all lemmas have been treated.</p> <p>If you would prefer some other style, you could put something like this in the preamble:</p> <pre>\ifledfinal\else   \renewcommand*\showlemma[1]{\textbf{#1}}% or simply ... [1]{#1} \fi</pre>

### 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, say `\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 17 p. 29, and described in more detail on XII.6.3 p. 147, 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}`

### 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>22</sup> you will find  $\LaTeX$  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  $\LaTeX$  macro that takes two arguments and throws away the first one.) The first incantation lets color show in both the main text and footnotes whereas the second one shows color in the main text but kills it in the lemma and footnotes. On the other hand if you use `\textcolor` instead, like

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

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

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

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

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

---

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



## 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}[2015/08/26 v2.1.2 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 default. We use `xkeyval` in order to manage options with argument.

`\ifnocritical@`  
`\if@noeled@sec` `\RequirePackage{xkeyval}`  
`\ifnoend@` %

`\ifnofamiliar@`  
`\ifnoledgroup@` The `parledgroup` option is for `reledpar`. However, it has consequence on `reledmac` internal command. So we need to define the boolean now.

`\ifparapparatus@`  
`\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@`  
`\newif\if@noeled@sec%`  
`\DeclareOptionX{noeledsec}{\@noeled@sectrue}`

`\newif\ifnocritical@%`  
`\DeclareOptionX{nocritical}{\nocritical@true}%`

`\newif\ifnofamiliar@%`  
`\DeclareOptionX{nofamiliar}{\nofamiliar@true}%`

`\newif\ifnoledgroup@%`  
`\DeclareOptionX{noledgroup}{\noledgroup@true}%`

`\newif\ifnoend@%`  
`\DeclareOptionX{noend}{%`  
`\let\l@dend@open\@gobble%`  
`\let\l@dend@close\relax%`  
`\global\let\l@dend@stuff=\relax%`  
`\noend@true%`

`}%`  
`\newif\ifnoquotation@`  
`\DeclareOptionX{noquotation}{\noquotation@true}`

`\newif\ifledfinal`  
`\DeclareOptionX{final}{\ledfinaltrue}`  
`\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 for paragraphed notes.

```

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

```

```

74 \RequirePackage{etex}
75 \reserveinserts{32}
76 \RequirePackage{suffix}
77 \RequirePackage{xstring}
78 \RequirePackage{ifluatex}
79 \RequirePackage{ragged2e}
80 \RequirePackage{ragged2e}
81 \RequirePackage{ifxetex}%
82 %

```

## II.5 Boolean flags

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

```

83 \newif\ifl@dmemoir
84 \@ifclassloaded{memoir}{\l@dmemoirtrue}{\l@dmemoirfalse}
85
86 %

```

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

```

87 \newif\ifl@imakeidx
88 \@ifpackageloaded{imakeidx}{\l@imakeidxtrue}{}%False is the default value
89 %

```

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

```

90 \newif\ifl@indextools%
91 \@ifpackageloaded{indextools}{%
92   \l@indextoolstrue%
93   \l@imakeidxtrue%
94   \let\imki@wrindexentry\indtl@wrindexentry%
95 }{}%
96 %

```

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.

```

97 \ifdef{\if@RTL}{\newif\if@RTL}
98 %

```

## II.6 Messages

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

`\reledmac@warning` Write a warning message.

```
99 \newcommand{\reledmac@warning}[1]{\PackageWarning{reledmac}{#1}}
100 %
```

`\reledmac@error` Write an error message.

```
101 \newcommand{\reledmac@error}[2]{\PackageError{reledmac}{#1}{#2}}
102 %
```

`\led@err@NumberingStarted` `\newcommand*{\led@err@NumberingStarted}{%`

`\led@err@NumberingNotStarted` `\reledmac@error{Numbering has already been started}{\@ehc}}`

`\led@err@NumberingShouldHaveStarted` `\newcommand*{\led@err@NumberingNotStarted}{%`

```
106 \reledmac@error{Numbering was not started}{\@ehc}}
```

```
107 \newcommand*{\led@err@NumberingShouldHaveStarted}{%

```

```
108 \reledmac@error{Numbering should already have been started}{\@ehc}}
```

```
109 %
```

`\led@err@edtextoutsidestart` `\newcommand*{\led@err@edtextoutsidestart}{%`

```
111 \reledmac@error{\string\edtext\space outside numbered paragraph (\pstart\
ldots\pend)}{\@ehc}}%
```

```
112 %
```

`\led@mess@NotesChanged` `\newcommand*{\led@mess@NotesChanged}{%`

```
114 \typeout{reledmac reminder: }%
```

```
115 \typeout{ The number of the footnotes in this section
116 has changed since the last run.}%
```

```
117 \typeout{ You will need to run LaTeX two more times
118 before the footnote placement}%
```

```
119 \typeout{ and line numbering in this section are
120 correct.}}
```

```
121 %
```

`\led@mess@SectionContinued` `\newcommand*{\led@mess@SectionContinued}[1]{%`

```
123 \message{Section #1 (continuing the previous section)}
```

```
124 %
```

`\led@err@LineationInNumbered` `\newcommand*{\led@err@LineationInNumbered}{%`

```
126 \reledmac@error{You can't use \string\lineation\space within
127 a numbered section}{\@ehc}}
```

```
128 %
```

```

\led@warn@BadLineation29 \newcommand*{\led@warn@BadLineation}{%
\led@warn@BadLinenummargin30 \reledmac@warning{Bad \string\lineation\space argument}}
\led@warn@BadLockdisp31 \newcommand*{\led@warn@BadLinenummargin}{%
\led@warn@BadSublockdisp32 \reledmac@warning{Bad \string\linenummargin\space argument}}
133 \newcommand*{\led@warn@BadLockdisp}{%
134 \reledmac@warning{Bad \string\lockdisp\space argument}}
135 \newcommand*{\led@warn@BadSublockdisp}{%
136 \reledmac@warning{Bad \string\sublockdisp\space argument}}
137 %

\led@warn@NoLineFile38 \newcommand*{\led@warn@NoLineFile}[1]{%
139 \reledmac@warning{Can't find line-list file #1}}
140 %

\led@warn@LineFileObsolete41 \newcommand*{\led@warn@Obsolete}[1]{%
142 \reledmac@warning{Line-list file #1 was obsolete. We have not read it.
Please run LaTeX again.}}
143 %

\led@warn@BadAdvancelineSubline44 \newcommand*{\led@warn@BadAdvancelineSubline}{%
\led@warn@BadAdvancelineLine45 \reledmac@warning{\string\advanceline\space produced a sub-line
146 number less than zero.}}
147 \newcommand*{\led@warn@BadAdvancelineLine}{%
148 \reledmac@warning{\string\advanceline\space produced a line
149 number less than zero.}}
150 %

\led@warn@BadSetline51 \newcommand*{\led@warn@BadSetline}{%
\led@warn@BadSetlinenum52 \reledmac@warning{Bad \string\setline\space argument}}
153 \newcommand*{\led@warn@BadSetlinenum}{%
154 \reledmac@warning{Bad \string\setlinenum\space argument}}
155 %

\led@err@PstartNotNumbered56 \newcommand*{\led@err@PstartNotNumbered}{%
\led@err@PstartInPstart57 \reledmac@error{\string\pstart\space must be used within a
\led@err@PendNotNumbered58 numbered section}{\@ehc}}
\led@err@PendNoPstart59 \newcommand*{\led@err@PstartInPstart}{%
\led@err@AutoparNotNumbered60 \reledmac@error{\string\pstart\space encountered while another
\led@err@NumberingWithoutPstart61 \string\pstart\space was in effect}{\@ehc}}
162 \newcommand*{\led@err@PendNotNumbered}{%
163 \reledmac@error{\string\pend\space must be used within a
164 numbered section}{\@ehc}}
165 \newcommand*{\led@err@PendNoPstart}{%
166 \reledmac@error{\string\pend\space must follow a \string\pstart}{\@ehc}}

```

```

167 \newcommand*\led@err@AutoparNotNumbered}{%
168   \reledmac@error{\string\autopar\space must be used within a
169     numbered section}{\@ehc}}
170 \newcommand*\led@err@NumberingWithoutPstart}{%
171   \reledmac@error{\string\beginnumbering...\string\endnumbering\space
    without \string\pstart}{\@ehc}}%
172 %

```

```

\led@warn@BadAction 73 \newcommand*\led@warn@BadAction}{%
174   \reledmac@warning{Bad action code, value \next@action.}}
175 %

```

```

\led@warn@DuplicateLabel 76 \newcommand*\led@warn@DuplicateLabel}[1]{%
\led@warn@AppLabelOutEdtext 77   \reledmac@warning{Duplicate definition of label `#1' on page \the\pageno
\led@warn@RefUndefined .}}
178 \newcommand*\led@warn@AppLabelOutEdtext}[1]{%
179   \reledmac@warning{\string\applabel\space outside of \string\edtext\space
    `#1' on page \the\pageno.}}%
180 \newcommand*\led@warn@RefUndefined}[1]{%
181   \reledmac@warning{Reference `#1' on page \the\pageno\space undefined.
    Using `000'.}}
182 %
183 %

```

```

\led@warn@NoMarginpars 84 \newcommand*\led@warn@NoMarginpars}{%
185   \reledmac@warning{You can't use \string\marginpar\space in numbered text
    }}
186 %

```

```

\led@warn@BadSidenotemargin 87 \newcommand*\led@warn@BadSidenotemargin}{%
188   \reledmac@warning{Bad \string\sidenotemmargin\space argument}}
189 %

```

```

\led@warn@NoIndexFile 90 \newcommand*\led@warn@NoIndexFile}[1]{%
191   \reledmac@warning{Undefined index file #1}}
192 %

```

```

\led@warn@SeriesStillExist 93 \newcommand*\led@warn@SeriesStillExist}[1]{%
194   \reledmac@warning{Series #1 is still existing !}}%
195 }%
196 %

```



```

\led@err@ManySidenotes97 \newcommand{\led@err@ManySidenotes}{%
\led@err@ManyLeftnotes98   \ifledRcol{%
\led@err@ManyRightnotes99   \reledmac@warning{\itemcount@\space sidenotes on line \the\line@numR\
space p. \the\page@numR}%
    \else%
    \reledmac@warning{\itemcount@\space sidenotes on line \the\line@num\
space p. \the\page@num}%
    \fi%
  }%
\newcommand{\led@err@ManyLeftnotes}{%
    \ifledRcol{%
    \reledmac@warning{\itemcount@\space leftnotes on line \the\line@numR\
space p. \the\page@numR}%
    \else%
    \reledmac@warning{\itemcount@\space leftnotes on line \the\line@num\
space p. \the\page@num}%
    \fi%
  }%
\newcommand{\led@err@ManyRightnotes}{%
    \ifledRcol{%
    \reledmac@warning{\itemcount@\space rightnotes on line \the\line@numR\
space p. \the\page@numR}%
    \else%
    \reledmac@warning{\itemcount@\space rightnotes on line \the\line@num\
space p. \the\page@num}%
    \fi%
  }%
  }%
  %
218

\led@err@TooManyColumns19 \newcommand*\led@err@TooManyColumns{%
\led@err@UnequalColumns20   \reledmac@error{Too many columns}{\@ehc}}
\led@err@LowStartColumn21 \newcommand*\led@err@UnequalColumns{%
\led@err@HighEndColumn22   \reledmac@error{Number of columns is not equal to the number
\led@err@ReverseColumns23   in the previous row (or \protect\\ \space forgotten?)}{\@ehc}}
224 \newcommand*\led@err@LowStartColumn{%
225   \reledmac@error{Start column is too low}{\@ehc}}
226 \newcommand*\led@err@HighEndColumn{%
227   \reledmac@error{End column is too high}{\@ehc}}
228 \newcommand*\led@err@ReverseColumns{%
229   \reledmac@error{Start column is greater than end column}{\@ehc}}
230 %

err@EdtextWithoutFootnote31 \newcommand{\led@err@EdtextWithoutFootnote}{%
232   \reledmac@error{edtext without Xfootnote. Check syntaxis.}{\@ehc}%
233 }%
234 %

```

```

\led@err@FootnoteWithoutEdtext235 \newcommand{\led@err@FootnoteWithoutEdtext}{%
236 \reledmac@error{Xfootnote without edtext. Check syntax.}\@ehc}%
237 }%
238 %

```

```

\led@error@ImakeidxAfterEledmac239 \newcommand{\led@error@ImakeidxAfterEledmac}{%
240 \reledmac@error{Imakeidx must be loaded before reledmac.}\@ehc}%
241 }%
242 %

```

```

\led@error@IndextoolsAfterEledmac243 \newcommand{\led@error@IndextoolsAfterEledmac}{%
244 \reledmac@error{Indextools must be loaded before reledmac.}\@ehc}%
245 }%
246 %

```

```

\led@error@fail@patch@@makecol247 \newcommand{\led@error@fail@patch@@makecol}{%
248 \reledmac@error{Fail to patch \string\@makecol\space command.}\@ehc}%
249 }%
250 %

```

```

\led@error@fail@patch@@reinserts251 \newcommand{\led@error@fail@patch@@reinserts}{%
252 \reledmac@error{Fail to patch \string\@reinserts\space command.}\@ehc}%
253 }%
254 %

```

```

\led@error@fail@patch@@doclearpage255 \newcommand{\led@error@fail@patch@@doclearpage}{%
256 \reledmac@error{Fail to patch \string\@doclearpage\space command.}\@ehc}%
257 }%
258 %

```

```

\led@error@fail@patch@@iiiminipage259 \newcommand{\led@error@fail@patch@@iiiminipage}{%
260 \reledmac@error{Fail to patch \string\@iiiminipage\space command.}\@ehc}%
261 }%
262 %

```

```

\led@error@fail@patch@endminipage263 \newcommand{\led@error@fail@patch@endminipage}{%
264 \reledmac@error{Fail to patch \string@endminipage\space command.}\@ehc}%
265 }%
266 %

```

## II.7 Gobbling

Here, we define some commands which gobble their arguments.

```
\@gobblethree67 \providecommand*\@gobblethree}[3]{}
\@gobblefour68 \providecommand*\@gobblefour}[4]{}
\@gobblefive69 \providecommand*\@gobblefive}[5]{}
270 %
```

## II.8 Miscellaneous commands

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

```
271 \ifl@final
272   \newcommand*\showlemma[1]{#1}
273 \else
274   \newcommand*\showlemma[1]{\underline{#1}}
275 \fi
276
277 %
```

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

Initialize it as `\empty`.

```
278 \let\linenumberlist=\empty
279
280 %
```

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

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

```
281 \newcount\@l@tempcnta \newcount\@l@tempcntb
282 %
```

## II.9 Prepare reledpar

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

`\ifl@pairing`

`\ifl@dpaging`

`\l@dpagingtrue`

`\l@dpagingfalse`

`\ifl@printingpages`

`\l@printingpagestrue`

`\l@printingpagesfalse`

`\ifl@printingcolumns`

`\l@printingcolumnstrue`

`\l@printingcolumnsfalse`

`\l@dpairingtrue`

`\l@dpairingfalse`

`\ifpst@rtedL`

`\pst@rtedLtrue`

`\pst@rtedLfalse`

`\l@dnumpstartsL`

`\ifledRcol`

```
283 \newif\ifl@pairing
284 \newif\ifl@dpaging%
285 \newif\ifl@printingpages%
286 \newif\ifl@printingcolumns%
287 \newif\ifpst@rtedL
```

```
288 \newcount\l@dnumpstartsL
289 %
```

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

```
290 \newif\ifledRcol
291 \newif\ifledRcol@
292 %
```

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

```
293 \newif\ifnumberingR
294 %
```

### III Sectioning commands

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

**`\extensionchars`** Each section will read and write an associated 'line-list file', containing information used to do the numbering; the file will be called `\jobname\section@num`, 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.

```
295 \newcount\section@num
296 \section@num=0
297 \let\extensionchars=\empty
298 %
```

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

**`\numberingtrue`**  
**`\numberingfalse`**

```
299 \newif\ifnumbering
300 %
```

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

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

- zero `\l@dnumpstartsL` — the number of chunks to be processed.
- set `\ifpst@rtedL` to FALSE.

```

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

```

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.

```

317 \else
318   \begingroup
319   \global\@afterindenttrue%In order to reestablish normal feature if the \
  begingroup was not here
320   \initnumbering@quote
321   \ifwidthliketwocolumns%
322     \csuse{setwidthliketwocolumns@\columns@position}%
323     \csuse{setpositionliketwocolumns@\columns@position}%
324   \fi%

```

```

325 \fi
326 \gdef\eled@sections@@{}%
327 \if@noeled@sec\else%
328 \makeatletter\inputIfFileExists{\jobname.eledsec\the\section@num}{-}{-}\
makeatother%
329 \immediate\openout\eled@sectioning@out=\jobname.eledsec\the\section@num
\relax%
330 \fi%
331 }
332 \newcommand*\initnumbering@reg{%
333 \global\pst@rtedLfalse
334 \global\l@dnumpststartsL \z@
335 \global\absline@num \z@
336 \gdef\normal@page@break{}
337 \gdef\l@prev@pb{}
338 \gdef\l@prev@nopb{}
339 \global\line@num \z@
340 \global\subline@num \z@
341 \global\@lock \z@
342 \global\sub@lock \z@
343 \global\sublines@false
344 \global\let\next@page@num=\relax
345 \global\let\sub@change=\relax
346 \resetprevline@
347 \resetprevpage@num
348 }
349
350 %

```

`\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.

```

351 \def\endnumbering{%
352 \ifnumbering
353 \global\numberingfalse
354 \normal@pars
355 \ifnum\l@dnumpststartsL=0%
356 \led@err@NumberingWithoutPstart%
357 \fi%
358 \ifl@dpairing
359 \global\pst@rtedLfalse
360 \else
361 \ifx\insertlines@list\empty\else
362 \global\noteschanged@true
363 \fi
364 \ifx\line@list\empty\else
365 \global\noteschanged@true
366 \fi
367 \fi

```

```

368 \ifnoteschanged@
369 \led@mess@NotesChanged
370 \fi
371 \else
372 \led@err@NumberingNotStarted
373 \fi
374 \autoparfalse
375 \if@noeled@sec\else%
376 \immediate\closeout\eled@sectioning@out%
377 \fi%
378 \ifl@dpairing\else
379 \global\l@dnumstartsL=\z@%
380 \endgroup
381 \fi
382 }
383 %

```

`\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>23</sup>

`\resumenumbering`

```

384 \newcommand{\pausenumbering}{%
385 \ifautopar\global\autopar@pausetrue\fi%
386 \endnumbering\global\numberingtrue}
387 %

```

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.

```

388 \newcommand*{\resumenumbering}{%
389 \ifnumbering
390 \ifautopar@pause\autopar\fi
391 \global\pst@rtedLtrue
392 \global\advance\section@num \@ne
393 \led@mess@SectionContinued{\the\section@num}%
394 \line@list@stuff{\jobname.\extensionchars\the\section@num}%
395 \l@dend@stuff
396 \ifl@dpairing\else%
397 \begingroup%
398 \initnumbering@quote%
399 \ifwidthliketwocolumns%
400 \csuse{setwidthliketwocolumns@\columns@position}%
401 \csuse{setpositionliketwocolumns@\columns@position}%
402 \fi%
403 \fi%
404 \else
405 \led@err@NumberingShouldHaveStarted
406 \endnumbering

```

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

```

407 \beginnumbering
408 \fi}
409
410
411 %

```

## 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  $\text{\LaTeX}$  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.

```

412 \newcommand*\list@create}[1]{%
413   \global\let#1=\empty%
414 }%
415 %

```

**\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 .

```

416 \newcommand*\list@clear}[1]{%
417   \global\let#1=\empty%
418 }
419 %

```

**\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`.

```

420 \newtoks\led@toksa \newtoks\led@toksb
421 \global\led@toksa={\}
422 \long\def\xright@appenditem#1\to#2{%
423   \global\led@toksb=\expandafter{#2}%

```



```

424 \xdef#2{\the\led@toksb\the\led@toksa\expandafter{#1}}%
425 \global\led@toksb={}}
426 %

```

**\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.

```

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

```

**\gl@p** The \gl@p macro removes the leftmost item from a list and places it in a control sequence. You say \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.

```

432 \def\gl@p#1\to#2{\expandafter\gl@poff#1\gl@poff#1#2}
433 \long\def\gl@poff\#1#2\gl@poff#3#4{\gdef#4{#1}\gdef#3{#2}}
434
435 %

```

## 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.

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

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

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

```

436 \newif\ifbypage@
437 \newif\ifbypstart@
438 %

```

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@R39 \newif\ifbypage@R
\ifbypstart@R40 \newif\ifbypstart@R
441 %

```

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

```
442 \newcommand*{\lineation}[1]{%
443 %
```

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

```
444 \ifnumbering
445 \led@err@LineationInNumbered
446 \else
447 %
```

If the argument is page.

```
448 \def\@tempa{#1}\def\@tempb{page}%
449 \ifx\@tempa\@tempb
450 \global\bypage@true
451 \global\bypstart@false
452 \unless\ifnocritical@%
453 \Xpstart[] [false]%
454 \fi%
455 %
```

If the argument is pstart.

```
456 \else
457 \def\@tempb{pstart}%
458 \ifx\@tempa\@tempb
459 \global\bypage@false
460 \global\bypstart@true
461 \unless\ifnocritical@%
462 \Xpstart%
463 \fi%
464 %
```

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

```
465 \else
466 \def\@tempb{section}
467 \ifx\@tempa\@tempb
468 \global\bypage@false
469 \global\bypstart@false
470 \unless\ifnocritical@%
471 \Xpstart[] [false]%
472 \fi%
473 %
```

In other case, it is an error.

```
474 \else
475 \led@warn@BadLineation
476 \fi
477 \fi
478 \fi
```

```

479 \fi}}
480 %

```

## V.2 Line number margin

`\linenummargin` `\linenummargin{⟨word⟩}` specify which margin line numbers are in; it takes one argument, a string, which value can be left ; right; inner or outer.

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

`\l@dgetline@margin`

```

481 \newcount\line@margin
482
483 \newcommand*{\linenummargin}[1]{%
484   \l@dgetline@margin{#1}%
485   \ifnum\@l@dttempcntb>\m@ne
486     \ifledRcol
487       \global\line@marginR=\@l@dttempcntb
488     \else
489       \global\line@margin=\@l@dttempcntb
490     \fi
491   \fi}}
492
493 \newcommand*{\l@dgetline@margin}[1]{%
494   \def\@tempa{#1}\def\@tempb{left}%
495   \ifx\@tempa\@tempb
496     \@l@dttempcntb \z@
497   \else
498     \def\@tempb{right}%
499     \ifx\@tempa\@tempb
500       \@l@dttempcntb \@ne
501     \else
502       \def\@tempb{outer}%
503       \ifx\@tempa\@tempb
504         \@l@dttempcntb \tw@
505       \else
506         \def\@tempb{inner}%
507         \ifx\@tempa\@tempb
508           \@l@dttempcntb \thr@@
509         \else
510           \led@warn@BadLinenummargin
511           \@l@dttempcntb \m@ne
512         \fi
513       \fi
514     \fi
515   \fi}
516
517 %

```

### 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.

```
518 \newcounter{firstlinenum}
519   \setcounter{firstlinenum}{5}
520 \newcounter{linenumincrement}
521   \setcounter{linenumincrement}{5}
522 %
```

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

```
523 \newcounter{firstsublinenum}
524   \setcounter{firstsublinenum}{5}
525 \newcounter{sublinenumincrement}
526   \setcounter{sublinenumincrement}{5}
527
528 %
```

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

```
529 \newcommand*{\firstlinenum}[1]{%
530   \ifledRcol%
531     \setcounter{firstlinenumR}{#1}%
532   \else%
533     \setcounter{firstlinenum}{#1}%
534   \fi%
535 }
536 \newcommand*{\linenumincrement}[1]{%
537   \ifledRcol%
538     \setcounter{linenumincrementR}{#1}%
539   \else%
540     \setcounter{linenumincrement}{#1}%
541   \fi%
542 }
543 \newcommand*{\firstsublinenum}[1]{%
544   \ifledRcol%
545     \setcounter{firstsublinenumR}{#1}%
546   \else%
547     \setcounter{firstsublinenum}{#1}%
548   \fi%
549 }
550 \newcommand*{\sublinenumincrement}[1]{%
551   \ifledRcol%
```

```

553 \setcounter{sublinenumincrementR}{#1}%
554 \else%
555 \setcounter{sublinenumincrement}{#1}%
556 \fi%
557 }
558
559 %

```

## 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.

```

560 \newcount\lock@disp
561 \newcommand{\lockdisp}[1]{%
562 \l@getlock@disp{#1}%
563 \ifnum\l@dttempcntb>\m@ne
564 \global\lock@disp=\l@dttempcntb
565 \else
566 \led@warn@BadLockdisp
567 \fi}}
568 \newcommand*{\l@getlock@disp}[1]{
569 \def\@tempa{#1}\def\@tempb{first}%
570 \ifx\@tempa\@tempb
571 \l@dttempcntb \z@
572 \else
573 \def\@tempb{last}%
574 \ifx\@tempa\@tempb
575 \l@dttempcntb \@ne
576 \else
577 \def\@tempb{all}%
578 \ifx\@tempa\@tempb
579 \l@dttempcntb \tw@
580 \else
581 \l@dttempcntb \m@ne
582 \fi
583 \fi
584 \fi}
585
586 %

```

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

```

587 \newcount\sublock@disp
588 \newcommand{\sublockdisp}[1]{%

```

```

589 \l@getlock@disp{#1}%
590 \ifnum\l@dttempcntb>\m@ne
591   \global\sublock@disp=\l@dttempcntb
592 \else
593   \led@warn@BadSublockdisp
594 \fi}}
595
596 %

```

## V.5 Line number style

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

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

`\sublinenumrep` and `\sublinenumr@p`.

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

`\sublinenumr@p`

```

597 \newcommand*{\linenumberstyle}[1]{%
598   \def\linenumrep##1{\@nameuse{#1}{##1}}}
599 \newcommand*{\sublinenumberstyle}[1]{%
600   \def\sublinenumrep##1{\@nameuse{#1}{##1}}}
601 %

```

Initialise the number styles to arabic.

```

602 \linenumberstyle{arabic}
603 \let\linenumr@p\linenumrep
604 \sublinenumberstyle{arabic}
605 \let\sublinenumr@p\sublinenumrep
606
607 %

```

## V.6 Line number printing

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

`\rightlinenum`

`\linenumsep`

`\numlabfont`

`\ledlinenum`

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

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

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

```

608 \newlength{\linenumsep}
609 \setlength{\linenumsep}{1pc}
610 \newcommand*{\numlabfont}{\normalfont\scriptsize}
611 \newcommand*{\ledlinenum}{%
612   \bgroup%
613   \ifluatex%
614     \luatextextdir TLT%
615   \fi%
616   \numlabfont\linenumrep{\line@num}%
617   \ifsublines@
618     \ifnum\subline@num>0\relax
619       \unskip\fullstop\sublinenumrep{\subline@num}%
620     \fi
621   \fi%
622   \egroup%
623 }%
624
625 \newcommand*{\leftlinenum}{%
626   \ledlinenum
627   \kern\linenumsep}
628 \newcommand*{\rightlinenum}{%
629   \kern\linenumsep
630   \ledlinenum}
631
632 %

```

## V.7 Line number counters and lists

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

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

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

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

```

633 \newcount\line@num

```

634 %

`\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.

635 `\newcount\subline@num`  
636 %

`\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`

637 `\newif\ifsublines@`  
638 %

`\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.

639 `\newcount\absline@num`  
640 %

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.

641 `\newcount\@lock`  
642 `\newcount\sub@lock`  
643 %



## V.9 Line number associated to lemma

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

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

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

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

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

```

644 \list@create{\line@list}
645 \list@create{\insertlines@list}
646 \list@create{\actionlines@list}

```

```

647 \list@create{\actions@list}
648
649 %

```

`\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`

```

650 \newcount\page@num
651 \newcount\endpage@num
652 \newcount\endline@num
653 \newcount\endsubline@num
654 %

```

`\ifnoteschanged@` If the number of the footnotes in a section is different from what it was during the last run, or if this is the very first time you've run  $\LaTeX$ , on this file, the information from the line-list used to place the notes will be wrong, and some notes will probably be misplaced. When this happens, we prefer to give a single error message for the whole section rather than messages at every point where we notice the problem, because we do not really know where in the section notes were added or removed, and the solution in any case is simply to run  $\LaTeX$  two more times; there is no fix needed to the document. The `\ifnoteschanged@` flag is set if such a change in the number of notes is discovered at any point.

`\noteschanged@true`

`\noteschanged@false`

```

655 \newif\ifnoteschanged@
656 %

```

`\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@ 657 \newcommand*{\resetprevline@}{%
658   \def\do##1{\global\csundef{prevline##1}}%
659   \dolistloop{\@series}%
660 }
661 %

```

`\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@ 662 \newcommand*{\resetprevpage@num}{%
663   \def\do##1{\ifcsdef{prevpage##1@num}{\global\csname prevpage##1@num\endcsname=0}{}}%

```

```

664 \dolistloop{\@series}%
665 }
666 %

```

## 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.

```

667 \newread\@inputcheck
668 \newcommand*{\read@linelist}[1]{%
669   \ifledRcol%
670     \list@clearing@regR%
671   \else%
672     \list@clearing@reg%
673   \fi%
674 %

```

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).

```

675   \list@clear{\maxlinesinpar@list}
676 %

```

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

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

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

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

```

677 \get@linelistfile{#1}%
678 \endgroup
679 %

```

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.

```

680 \ifledRcol
681   \global\page@numR=\m@ne
682   \ifx\actionlines@listR\empty
683     \gdef\next@actionlineR{1000000}%
684   \else
685     \gl@p\actionlines@listR\to\next@actionlineR
686     \gl@p\actions@listR\to\next@actionR
687   \fi
688 \else
689   \global\page@num=\m@ne
690   \ifx\actionlines@list\empty
691     \gdef\next@actionline{1000000}%
692   \else
693     \gl@p\actionlines@list\to\next@actionline
694     \gl@p\actions@list\to\next@action
695   \fi
696 \fi
697 }
698 %

```

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

```

699 \newcommand*{\list@clearing@reg}{%
700   \list@clear{\line@list}%
701   \list@clear{\insertlines@list}%
702   \list@clear{\actionlines@list}%
703   \list@clear{\actions@list}%
704   \list@clear{\linesinpar@listL}%
705   \list@clear{\linesonpage@listL}%
706   }%
707 %

```

`\get@linelistfile` reledmac can take advantage of the L<sup>A</sup>T<sub>E</sub>X ‘safe file input’ macros to get the line-list file.

```

708 \newcommand*{\get@linelistfile}[1]{%
709   \InputIfFileExists{#1}{%
710     \global\noteschanged@false
711     \begingroup
712       \catcode`\[=1 \catcode`\]=2
713       \makeatletter \catcode`\^^M=9}{%
714     \led@warn@NoLineFile{#1}%
715     \global\noteschanged@true
716     \begingroup}%
717   }
718 %
719 %

```

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  $\text{\LaTeX}$  for that to work. We have retained this slower approach to avoid that sort of hacking about, but have provided the `\pausenumbering` and `\resumenumbering` macros to help you if you run into macro memory limitations (see 4.2.7 p. 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 say `\global`. This is because we want them to affect only the counter values within the current group when nested calls of `\@ref` occur. (The code assumes throughout that the value of `\globaldefs` is zero.)

The macros with action in their names contain all the code that modifies the action-code list: again, this is so that they can be turned off easily for nested calls of `\@ref`.

`\line@list@version` The `\line@list@version` check if the line-list file does not refers to the older commands of `reledmac`. In this case, we stop reading the line-list file. Consequently, `\line@list@version` must be the first line of a line-number file.

```

720 \newcommand{\line@list@version}[1]{%
721   \IfStrEq{#1}{\this@line@list@version}%
722   {}%
723   {\ifledRcol%
724     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
725     \else%
726     \led@warn@Obsolete{\jobname.\extensionchars\the\section@num}%
727     \fi%
728     \endinput%
729   }%
730 }%
731 %

```

`\@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`.

```

732
733 \newcommand*{\@nl}[2]{%
734   \fix@page{#1}%
735   \ifledRcol%
736     \@nl@regR%
737   \else%
738     \@nl@reg%
739   \fi%
740 }
741 \newcommand*{\@nl@reg}{%
742   \ifx\l@dchset@num\relax \else
743     \advance\absline@num \@ne
744     \set@line@action
745     \let\l@dchset@num=\relax
746     \advance\absline@num \m@ne
747     \advance\line@num \m@ne
748   \fi
749   %

```

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

```

750   \advance\absline@num \@ne
751     \ifx\next@page@num\relax \else
752       \page@action
753       \let\next@page@num=\relax
754     \fi
755     \ifx\sub@change\relax \else
756       \ifnum\sub@change>\z@
757         \sublines@true
758       \else
759         \sublines@false
760       \fi
761       \sub@action
762       \let\sub@change=\relax
763     \fi
764   %

```

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

```

765   \ifcase\@lock
766     \or

```

```

767         \@lock \tw@
768         \or \or
769         \@lock \z@
770     \fi
771     \ifcase\sub@lock
772         \or
773         \sub@lock \tw@
774         \or \or
775         \sub@lock \z@
776     \fi
777 %

```

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

```

778     \ifsublines@
779         \ifnum\sub@lock<\tw@
780             \advance\subline@num \@ne
781         \fi
782     \else
783         \ifnum\@lock<\tw@
784             \advance\line@num \@ne \subline@num \z@
785         \fi
786     \fi}
787
788 %

```

`\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`.

```

789 \newcount\last@page@num
790 \last@page@num=-10000
791
792 \newcommand*{\fix@page}[1]{%
793     \ifledRcol
794         \ifnum #1=\last@page@numR
795         \else
796             \ifbypage@R
797                 \line@numR \z@ \subline@numR \z@
798             \fi
799             \page@numR=#1\relax
800             \last@page@numR=#1\relax
801             \def\next@page@numR{#1}%
802         \fi
803     \else
804         \ifnum #1=\last@page@num
805         \else
806             \ifbypage@
807                 \line@num \z@ \subline@num \z@
808             \fi
809             \page@num=#1\relax
810             \last@page@num=#1\relax

```



```

811 \def\next@page@num{#1}%
812 \listxadd{\normal@page@break}{\the\absline@num}
813 \fi
814 \fi}
815 %

```

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

**\@lopL** **\@lopR**

```

816 \newcommand*\@pend}[1]{}
817 \newcommand*\@pendR}[1]{}
818 \newcommand*\@lopL}[1]{}
819 \newcommand*\@lopR}[1]{}
820
821 %

```

**\sub@on** **\sub@off** The `\sub@on` and `\sub@off` macros turn sub-lineation on and off: but not directly, since such changes do not really take effect until the next line of text. Instead they set a flag that notifies `\@n1` of the necessary action.

```

822 \newcommand*\sub@on{\ifsublines@
823 \let\sub@change=\relax
824 \else
825 \def\sub@change{1}%
826 \fi}
827 \newcommand*\sub@off{\ifsublines@
828 \def\sub@change{-1}%
829 \else
830 \let\sub@change=\relax
831 \fi}
832
833 %

```

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

```

834
835 \newcommand*\@adv}[1]{%
836 \ifsublines@
837 \ifledRcol
838 \advance\subline@numR by #1\relax
839 \ifnum\subline@numR<\z@
840 \led@warn@BadAdvancelineSubline
841 \subline@numR \z@
842 \fi
843 \else
844 \advance\subline@num by #1\relax
845 \ifnum\subline@num<\z@
846 \led@warn@BadAdvancelineSubline

```

```

847     \subline@num \z@
848     \fi
849   \fi
850 \else
851   \ifledRcol
852     \advance\line@numR by #1\relax
853     \ifnum\line@numR<\z@
854       \led@warn@BadAdvancelineLine
855       \line@numR \z@
856     \fi
857   \else
858     \advance\line@num by #1\relax
859     \ifnum\line@num<\z@
860       \led@warn@BadAdvancelineLine
861       \line@num \z@
862     \fi
863   \fi
864 \fi
865 \set@line@action}
866
867 %

```

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

```

868
869 \newcommand*{\@set}[1]{%
870   \ifledRcol
871     \ifsublines@
872       \subline@numR=#1\relax
873     \else
874       \line@numR=#1\relax
875     \fi
876     \set@line@action
877   \else
878     \ifsublines@
879       \subline@num=#1\relax
880     \else
881       \line@num=#1\relax
882     \fi
883     \set@line@action
884   \fi}
885
886 %

```

**\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.

```

887 \newcommand*{\l@d@set}[1]{%
888   \ifledRcol
889     \line@numR=#1\relax
890     \advance\line@numR \@ne
891     \def\l@dchset@num{#1}
892   \else
893     \line@num=#1\relax
894     \advance\line@num \@ne
895     \def\l@dchset@num{#1}
896   \fi}
897 \let\l@dchset@num\relax
898
899 %
900 %

```

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

```

901 \newcommand*{\page@action}{%
902   \ifledRcol
903     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
904     \xright@appenditem{\next@page@numR}\to\actions@listR
905   \else
906     \xright@appenditem{\the\absline@num}\to\actionlines@list
907     \xright@appenditem{\next@page@num}\to\actions@list
908   \fi}
909 %
910 %

```

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

```

911 \newcommand*{\set@line@action}{%
912   \ifledRcol
913     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
914     \ifsublines@
915       \@l@tempcnta=-\subline@numR
916     \else
917       \@l@tempcnta=-\line@numR
918     \fi
919     \advance\@l@tempcnta by -5000\relax
920     \xright@appenditem{\the\@l@tempcnta}\to\actions@listR
921   \else
922     \xright@appenditem{\the\absline@num}\to\actionlines@list
923     \ifsublines@
924       \@l@tempcnta=-\subline@num
925     \else
926       \@l@tempcnta=-\line@num
927     \fi
928     \advance\@l@tempcnta by -5000\relax
929

```

```

930 \xright@appenditem{\the\@l@dttempcnta}\to\actions@list
931 \fi}
932 %

```

**\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.

```

933
934 \newcommand*{\sub@action}{%
935   \ifl@edRcol
936     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
937     \ifsublines@
938       \xright@appenditem{-1001}\to\actions@listR
939     \else
940       \xright@appenditem{-1002}\to\actions@listR
941     \fi
942   \else
943     \xright@appenditem{\the\absline@num}\to\actionlines@list
944     \ifsublines@
945       \xright@appenditem{-1001}\to\actions@list
946     \else
947       \xright@appenditem{-1002}\to\actions@list
948     \fi
949   \fi}
950 %

```

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

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

```

951 \newcommand*{\lock@on}{\futurelet\next\do@lockon}
952
953 \newcommand*{\do@lockon}{%
954   \ifx\next\lock@off
955     \global\let\lock@off=\skip@lockoff
956   \else
957     \ifl@edRcol
958       \do@lockonR
959     \else
960       \do@lockonL
961     \fi
962   \fi}
963
964
965 \newcommand*{\do@lockonL}{%

```

```

966 \xright@appenditem{\the\absline@num}\to\actionlines@list
967 \ifsublines@
968   \xright@appenditem{-1005}\to\actions@list
969   \ifnum\sub@lock=\z@
970     \sub@lock \@ne
971   \else
972     \ifnum\sub@lock=\thr@@
973       \sub@lock \@ne
974     \fi
975   \fi
976 \else
977   \xright@appenditem{-1003}\to\actions@list
978   \ifnum\@lock=\z@
979     \@lock \@ne
980   \else
981     \ifnum\@lock=\thr@@
982       \@lock \@ne
983     \fi
984   \fi
985 \fi}
986
987 %/

```

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

```

\do@lockoff
\do@lockoffL
\skip@lockoff
988 \newcommand*{\do@lockoffL}{%
989   \xright@appenditem{\the\absline@num}\to\actionlines@list
990   \ifsublines@
991     \xright@appenditem{-1006}\to\actions@list
992     \ifnum\sub@lock=\tw@
993       \sub@lock \thr@@
994     \else
995       \sub@lock \z@
996     \fi
997   \else
998     \xright@appenditem{-1004}\to\actions@list
999     \ifnum\@lock=\tw@
1000       \@lock \thr@@
1001     \else
1002       \@lock \z@
1003     \fi
1004   \fi}
1005
1006 \newcommand*{\do@lockoff}{%
1007   \ifledRcol
1008     \do@lockoffR
1009   \else
1010     \do@lockoffL
1011   \fi}
1012 \newcommand*{\skip@lockoff}{\global\let\lock@off=\do@lockoff}

```

```

1013 \global\let\lock@off=\do@lockoff
1014
1015 %

```

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

```

1016 \newcommand*{\n@num}{%
1017   \ifledRcol%
1018     \xright@appenditem{\the\absline@numR}\to\actionlines@listR
1019     \xright@appenditem{-1007}\to\actions@listR
1020   \else%
1021     \xright@appenditem{\the\absline@num}\to\actionlines@list%
1022     \xright@appenditem{-1007}\to\actions@list%
1023   \fi%
1024 }%
1025
1026 %

```

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

```

1027 \newcommand*{\n@num@stanza}{%
1028   \ifledRcol%
1029     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1030     \xright@appenditem{-1008}\to\actions@listR%
1031   \else%
1032     \xright@appenditem{\the\absline@num}\to\actionlines@list%%
1033     \xright@appenditem{-1008}\to\actions@list%
1034   \fi%
1035 }
1036 %

```

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

**\hidenum**

```

1037 \newif\ifl@dhidenumber
1038 \newcommand*{\hidenum}{
1039   \ifledRcol%
1040     \write\linenum@outR{\string\hide@num}%
1041   \else%
1042     \write\linenum@out{\string\hide@num}%
1043   \fi%
1044 }%
1045 \newcommand*{\hide@num}{%
1046   \ifledRcol%
1047     \xright@appenditem{\the\absline@numR}\to\actionlines@listR%
1048     \xright@appenditem{-1009}\to\actions@listR%
1049   \else%
1050     \xright@appenditem{\the\absline@num}\to\actionlines@list%%

```

```

1051 \xright@appenditem{-1009}\to\actions@list%
1052 \fi%
1053 }
1054 %

```

**\@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.

```

1055 \newcount\insert@count
1056 %

```

- #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.

```

1057 \newcommand*\dummy@ref}[2]{#2}
1058 %

```

**\@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.

```

1059 \newcommand*\@ref}[2]{%
1060 \ifledRcol%
1061 \@ref@regR{#1}{#2}%
1062 \else%
1063 \@ref@reg{#1}{#2}%
1064 \fi%
1065 }%
1066 \newcommand*\@ref@reg}[2]{%
1067 \global\insert@count=#1\relax
1068 \global\advance\@edtext@level by 1%
1069 \loop\ifnum\insert@count>\z@
1070 \xright@appenditem{\the\absline@num}\to\insertlines@list
1071 \global\advance\insert@count \m@ne
1072 \repeat
1073 %

```

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.

```

1074 \begingroup
1075   \let\@ref=\dummy@ref
1076   \let\@lopL\@gobble
1077   \let\page@action=\relax
1078   \let\sub@action=\relax
1079   \let\set@line@action=\relax
1080   \let\@lab=\relax
1081   \let\@lemma=\relax%
1082   \let\@sw\@gobblethree%
1083   #2
1084   \global\endpage@num=\page@num
1085   \global\endline@num=\line@num
1086   \global\endsubline@num=\subline@num
1087 \endgroup
1088 %

```

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

```

1089   \xright@appenditem%
1090     {\the\page@num|\the\line@num|%
1091     \ifsublines@ \the\subline@num \else 0\fi|%
1092     \the\endpage@num|\the\endline@num|%
1093     \ifsublines@ \the\endsubline@num \else 0\fi}\to\line@list
1094 %

```

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.

```

1095   \expandafter\list@create\expandafter{\csname sw@list@edtext@tmp@\the\
1096   @edtext@level\endcsname}%
1097   \providebool{lemmacommand@\the\@edtext@level}%
1098   \boolfalse{lemmacommand@\the\@edtext@level}%
1099 %

```

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

```

1099   #2%
1100 %

```

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.

```

1101   \ifnum\@edtext@level>0%
1102     \def\create@this@edtext@level{\expandafter\list@create\expandafter{\
1103     csname sw@list@edtext@\the\@edtext@level\endcsname}}%
1104     \ifcsundef{sw@list@edtext@\the\@edtext@level}\create@this@edtext@level
1105     {}%
1106     \letcs{\@tmp}{sw@list@edtext@\the\@edtext@level}%
1107     \letcs{\@tmpp}{sw@list@edtext@tmp@\the\@edtext@level}%
1108     \xright@appenditem{\expandonce\@tmpp}\to\@tmp%
1109     \global\cslet{sw@list@edtext@\the\@edtext@level}{\@tmp}%

```



```

1108 \fi%
1109 %

Decrease edtext level counter.

1110 \global\advance\@edtext@level by -1%
1111 %

1112 }
1113
1114 %

```

## 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`.

```

1115 \newwrite\linenum@out
1116 %

```

`\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`.

```

1117 \newif\iffirst@linenum@out@
1118 \iffirst@linenum@out@true
1119 %

```

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

```

1120 \newcommand{\this@line@list@version}{2}%
1121 %

```

**\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.

```
1122 \newcommand*{\line@list@stuff}[1]{%
1123 %
```

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

```
1124 \read@linelist{#1}%
1125 %
```

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.

```
1126 \iffirst@linenum@out@
1127 \immediate\closeout\linenum@out%
1128 \global\first@linenum@out@false%
1129 \immediate\openout\linenum@out=#1\relax%
1130 \immediate\write\linenum@out{\string\line@list@version{\
this@line@list@version}}%
1131 \else
1132 %
```

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.

```
1133 \if@minipage%
1134 \leavevmode%
1135 \fi%
1136 \closeout\linenum@out%
1137 \openout\linenum@out=#1\relax%
1138 \fi}
1139
1140 %
```

**\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.

```
1141 \newcommand*{\new@line}{%
1142 \IfStrEq{\led@pb@setting}{after}%
1143 {\xifinlist{\the\absline@num}{\l@prev@nopb}%
1144 {\xifinlist{\the\absline@num}{\normal@page@break}%
1145 {\numgdef{\@next@page}{\c@page+1}%
1146 \write\linenum@out{\string\@nl[\@next@page][\@next@page]}%
1147 }%
1148 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}%
1149 }%
1150 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}%
1151 {}}%
```

```

1152 \IfStrEq{\led@pb@setting}{before}%
1153 {\numdef{\next@absline}{\the\absline@num+1}%
1154 \xifinlist{\next@absline}{\l@prev@nopb}%
1155 {\xifinlist{\the\absline@num}{\normal@page@break}%
1156 {\numgdef{\nc@page}{\c@page+1}%
1157 \write\linenum@out{\string\@nl[\nc@page][\nc@page]}}%
1158 }%
1159 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1160 }%
1161 {\write\linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1162 }%
1163 {}%
1164 \IfStrEqCase{\led@pb@setting}{\{before\}{\relax\}{after\}{\relax\}}{\write\
linenum@out{\string\@nl[\the\c@page][\thepage]}}%
1165 }
1166
1167 %

```

**\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**

```

1168 \newif\if@noneed@Footnote%
1169
1170 \newcommand*{\flag@start}{%
1171 \ifledRcol%
1172 \edef\next{\write\linenum@outR{%
1173 \string\@ref[\the\insert@countR] []}}%
1174 \next%
1175 \ifnum\insert@countR<1%
1176 \if@noneed@Footnote\else%
1177 \led@err@EdtextWithoutFootnote%
1178 \fi%
1179 \fi%
1180 \else%
1181 \edef\next{\write\linenum@out{%
1182 \string\@ref[\the\insert@count] []}}%
1183 \next%
1184 \ifnum\insert@count<1%
1185 \if@noneed@Footnote\else%
1186 \led@err@EdtextWithoutFootnote%
1187 \fi%
1188 \fi%
1189 \fi}%
1190
1191 %

```

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

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

```

1192
1193
1194 \newcommand*{\startsub}{\dimen0\lastskip
1195   \ifdim\dimen0>Opt \unskip \fi
1196   \ifledRcol \write\linenum@outR{\string\sub@on}%
1197   \else      \write\linenum@out{\string\sub@on}%
1198   \fi
1199   \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1200 \def\endsub{\dimen0\lastskip
1201   \ifdim\dimen0>Opt \unskip \fi
1202   \ifledRcol \write\linenum@outR{\string\sub@off}%
1203   \else      \write\linenum@out{\string\sub@off}%
1204   \fi
1205   \ifdim\dimen0>Opt \hskip\dimen0 \fi}
1206
1207 %

```

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

```

1208 \newcommand*{\advanceline}[1]{\leavevmode%
1209   \ifledRcol \write\linenum@outR{\string\@adv[#1]}%
1210   \else      \write\linenum@out{\string\@adv[#1]}%
1211   \fi}%
1212 }
1213 %

```

**\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.

```

1214
1215 \newcommand*{\setline}[1]{%
1216   \leavevmode%
1217   \ifnum#1<\z@
1218     \led@warn@BadSetline
1219   \else

```

```

1220 \ifledRcol \write\linenum@outR{\string\@set[#1]}%
1221 \else \write\linenum@out{\string\@set[#1]}%
1222 \fi
1223 \fi}
1224
1225 %

```

**\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.

```

1226
1227 \newcommand*{\setlinenum}[1]{%
1228 \ifnum#1<\z@
1229 \led@warn@BadSetlinenum
1230 \else
1231 \ifledRcol \write\linenum@outR{\string\l@d@set[#1]}
1232 \else \write\linenum@out{\string\l@d@set[#1]} \fi
1233 \fi}
1234
1235 %

```

**\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.

```

1236
1237 \newcommand*{\startlock}{%
1238 \ifledRcol \write\linenum@outR{\string\lock@on}%
1239 \else \write\linenum@out{\string\lock@on}%
1240 \fi}
1241 \def\endlock{%
1242 \ifledRcol \write\linenum@outR{\string\lock@off}%
1243 \else \write\linenum@out{\string\lock@off}%
1244 \fi}
1245 %

```

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

**\ifl@dskipversenumber**

**\l@dskipnumbertrue**

**\l@dskipnumberfalse**

**\skipnumbering**

```

1246 \newif\ifl@dskipnumber
1247 \newif\ifl@dskipversenumber%
1248 \newcommand*{\skipnumbering}{%
1249 \leavevmode%
1250 \ifledRcol%
1251 \ifinstanza%
1252 \write\linenum@outR{\string\n@num@stanza}%
1253 \else%
1254 \write\linenum@outR{\string\n@num}%
1255 \fi%
1256 \advanceline{-1}%

```

```

1257 \else%
1258   \ifistanza%
1259     \write\linenum@out{\string\n@num@stanza}%
1260   \else%
1261     \write\linenum@out{\string\n@num}%
1262   \fi%
1263   \advanceline{-1}%
1264 \fi%
1265 }%
1266
1267 %

```

## 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. 122). 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.

```
1268 \list@create{\end@lemmas}
1269 %
```

`\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.

```
1270 \newcommand{\dummy@edtext}[2]{#1}
1271 %
```

`\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`.

```

1272 \newcommand{\dummy@edtext@showlemma}[2]{\showlemma{#1}}%
1273 %

```

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

`\no@expands`  
`\morenoexpands`

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

The first class is font-changing macros. We suppress expansion for them by letting them become equal to zero.<sup>24</sup> This is done because we want to pass into our notes the generic commands to change to roman or whatever, and not their expansions that will ask for a particular style at a specified size. The notes may well be in a smaller font, so the command should be expanded later, when the note's environment is in effect.

A second sort to turn off includes a few of the accent macros. Most are not a problem: an accent that is expanded to an `\accent` command may be harder to read but it works just the same. The ones that cause problems are: those that use alignments— $\text{\TeX}$  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  $\text{\TeX}$`  has this sort of problem as well, but is not used enough to bother with. That macro, and any other that causes trouble, will get by all right if you put a `\protect` in front of it in your file.)

We also need to eliminate all `reledmac` macros like `\edlabel` and `\setline` that write things to auxiliary files: that writing should be done only once. And we make `\edtext` itself, if it appears within its own argument, do nothing but copy its first argument.

Finally, we execute `\morenoexpands`. The version of `\morenoexpands` defined here does nothing; but you may define a version of your own when you need to add more expansion suppressions as needed with your macros. That makes it possible to make such additions without needing to copy or modify the standard `reledmac` code. If you define your own `\morenoexpands`, you must be very careful about spaces: if the macro adds any spaces to the text when it runs, extra space will appear in the main text when `\edtext` is used.

(A related problem, not addressed by these two macros, is that of characters whose category code is changed by any the macros used in the arguments to `\edtext`. Since the category codes are set when the arguments are scanned, macros that depend on changing them will not work. We have most often encountered this with characters that are made 'active' within text in some, but not all, of the languages used within the document. One way around the problem, if it takes this form, is to ensure that those characters are *always* active; within languages that make no special use of them, their associated control sequences should simply return the proper character. A simpler solution is to avoid active character, using `Lua $\text{\TeX}$`  or `XY $\text{\LaTeX}$` .)

```

1274 \newcommand*{\no@expands}{%

```

<sup>24</sup>Since 'control sequences equivalent to characters are not expandable'—*The  $\text{\TeX}$ book*, answer to Exercise 20.14.



```

1275 \let\select@@lemmafont=0%
1276 \let\startsub=\relax \let\endsub=\relax
1277 \let\startlock=\relax \let\endlock=\relax
1278 \let\edlabel=\@gobble
1279 \let\setline=\@gobble \let\advanceline=\@gobble
1280 \let\sameword\sameword@inedtext%
1281 \let\edtext=\dummy@edtext
1282 \l@dtabnoexpands
1283 \morenoexpands}
1284 \let\morenoexpands=\relax
1285
1286 %

```

**\@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.

```

1287 \newcommand{\@tag}{}
1288 %

```

**\@edtext@level** This counter is increased by 1 at each level of \edtext. That is useful for some commands which can have a different behavior if called inside or outside of the  $\langle lemma \rangle$  argument.

```

1289 \newcount\@edtext@level%
1290 \@edtext@level=0%
1291 %

```

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

```

1292 \newcommand{\edtext}[2]{\leavevmode%
1293 %

```

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

```

1294 \ifnumberedpar%
1295 %

```

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

```

1296 \global\advance\@edtext@level by 1%
1297 %

```

By default, we do not use \lemma

```

1298 \global\@lemmacommand@false%
1299 %

```

```

1300 \begingroup%
1301 %

```

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.

```

1302     \ifledRcol%
1303         \ifcsundef{sw@list@edtextR@the\@edtext@level}%
1304             {\global\let\sw@inthisedtext\empty}%
1305             {\ifcsempy{sw@list@edtextR@the\@edtext@level}%
1306                 {\global\let\sw@inthisedtext\empty}%
1307                 {\expandafter\gl@p\csname sw@list@edtextR@the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1308             }%
1309         \else%
1310             \ifcsundef{sw@list@edtext@the\@edtext@level}%
1311                 {\global\let\sw@inthisedtext\empty}%
1312                 {\ifcsempy{sw@list@edtext@the\@edtext@level}%
1313                     {\global\let\sw@inthisedtext\empty}%
1314                     {\expandafter\gl@p\csname sw@list@edtext@the\@edtext@level\endcsname\
to\sw@inthisedtext}%
1315                 }%
1316             \fi%
1317 %

```

**\@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.

```

1318     \global\renewcommand{\@tag}{-%
1319         \no@expands #1%
1320     }%
1321 %

```

**\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`.

```

1322     \set@line%
1323 %

```

`\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`.

```

1324     \ifledRcol \global\insert@countR \z@%
1325     \else      \global\insert@count \z@ \fi%
1326 %

```

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.

```
1327 \ignorespaces #2\relax%
1328 %
```

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

```
1329 \@ifundefined{xpg@main@language}{%if not polyglossia
1330 \flag@start}%
1331 {\if@RTL\flag@end\else\flag@start\fi%
1332 }%
1333 %
```

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

```
1334 \if@lemmacommand%
1335 \ifledRcol%
1336 \write\linenum@outR{\string\@lemma}%
1337 \else%
1338 \write\linenum@out{\string\@lemma}%
1339 \fi%
1340 \fi%
1341 %
```

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.

```
1342 \endgroup%
1343 \showlemma{#1}%
1344 %
```

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.

```
1345 \ifx\end@lemmas\empty \else%
1346 \gl@p\end@lemmas\to\x@lemma%
1347 \x@lemma%
1348 \global\let\x@lemma=\relax%
1349 \fi%
1350 \@ifundefined{xpg@main@language}{%if not polyglossia
1351 \flag@end}%
1352 {\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).
1353 }%
1354 %
```

We switch to false some flags.

- 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`.

```

1355 \global\@noneed@Footnotefalse%
1356 \global\advance\@edtext@level by -1%
1357 \global\@lemmacommand@false%
1358 %

```

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

```

1359 \else%
1360 \showlemma{#1} (\textbf{\textsc{Edtext outside numbered paragraph}})\
led@err@edtextoutsidepstart%
1361 \fi%
1362 }%
1363
1364 \newcommand*{\flag@end}{%
1365 \ifledRcol%
1366 \write\linenum@outR{}}%
1367 \else%
1368 \write\linenum@out{}}%
1369 \fi}%
1370
1371 %

```

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

```

1372 \newif\ifnumberline
1373 \numberlinetrue
1374 %

```

`\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.

```

1375 \newcommand*{\set@line}{%
1376 \ifledRcol

```

```

1377 \ifx\line@listR\empty
1378 \global\noteschanged@true
1379 \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1380 \else
1381 \glp\line@listR\to\@tempb
1382 \xdef\l@d@nums{\@tempb|\edfont@info}%
1383 \global\let\@tempb=\undefined
1384 \fi
1385 \else
1386 \ifx\line@list\empty
1387 \global\noteschanged@true
1388 \xdef\l@d@nums{000|000|000|000|000|000|\edfont@info}%
1389 \else
1390 \glp\line@list\to\@tempb
1391 \xdef\l@d@nums{\@tempb|\edfont@info}%
1392 \global\let\@tempb=\undefined
1393 \fi
1394 \fi}
1395
1396 %

```

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

```

1397 \newcommand*\edfont@info{\f@encoding/\f@family/\f@series/\f@shape}
1398
1399 %

```

## 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. 106).

```

1400 \unless\ifnocritical@
1401 \newcommand*\lemma}[1]{%
1402 \global\@lemmacommand@true%
1403 \global\renewcommand{\@tag}{%
1404 \no@expands #1%
1405 }%
1406 \ignorespaces%
1407 }%
1408 %

```

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

```

1409 \newcommand{\@lemma}{%
1410 \booltrue{lemmacommand@the\@edtext@level}%
1411 }%

```

```

1412 \fi
1413 %

```

**\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.

```

1414 \newif\if@lemmacommand@%
1415 %

```

### 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. 81): 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.

```

1416 \newcommand*{\linenum}[1]{%
1417   \xdef\@tempa{#1|\\|\\|\\|\\|\\|\\|\\noexpand\\l@d@nums}%
1418   \global\let\l@d@nums=\empty
1419   \expandafter\line@set\@tempa|\\|ignorespaces}
1420 %

```

**\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.

```

1421 \def\line@set#1|#2\\#3|#4\\{%
1422   \gdef\@tempb{#1}%
1423   \ifx\@tempb\empty
1424     \l@d@add{#3}%
1425   \else
1426     \l@d@add{#1}%
1427   \fi
1428   \gdef\@tempb{#4}%
1429   \ifx\@tempb\empty\else
1430     \l@d@add{|}\line@set#2\\#4\\%
1431   \fi}
1432 %

```

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

```

1433 \newcommand{\l@d@add}[1]{\xdef\l@d@nums{\l@d@nums#1}}
1434
1435 %

```

## 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 `el@edmac` 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{\langle argument \rangle}+` is called from the beginning of the lineation to the end of the current line. We also store the same data for the preceding absolute line number, if it does not have `\sameword{\langle argument \rangle}`.
  - 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 `\sameword` which are called as lemma information, that is 1) or called in the first argument of `\sameword` 2) or called in the `\lemma` macro of the second argument of `\sameword` AND marked by the optional argument of `\sameword` in first argument of `\edtext`.

For example, suppose a line with nested `\edtexts` which contains some word marked by `\sameword` and having the following relative rank:

bar<sup>1</sup> foo<sup>1</sup> foo<sup>2</sup> bar<sup>2</sup> foo<sup>3</sup> (A)(B) foo<sup>4</sup> bar<sup>3</sup> (C) foo<sup>5</sup> (D) bar<sup>4</sup> (E)

In this example, all lemma information for `\edtext` is framed. The text in parenthesis is the content of critical notes associated to the preceding frame. As you can see, we have two level of `\edtext`.

The list for `\edtexts` of level 1 is  $\{\{1, 2, 2, 3, 4, 3\}, \{5, 4\}\}$ .

The list for `\edtexts` of level 2 is  $\{\{1, 2, 2, 3\}, \{5\}\}$ .

As you can see, the mandatory argument of `\sameword` does not matter: we store the rank informations for every word potentially ambiguous.

- At the second run, when a critical notes is called, we associate it to the next item of the list associated to is `\edtext` level. So, in the previous example:
  - Critical notes (A) and (B) are associated with  $\{1, 2, 2, 3\}$ .
  - Critical note (C) is associated with  $\{1, 2, 2, 3, 4, 3\}$ .
  - Critical note (D) is associated with  $\{5\}$ .
  - Critical note (E) is associated with  $\{5, 4\}$ .
- At the second run, when a critical note is printed:
  - The `\sameword` command is let `\sameword@inedtext`.
  - At each call of this `\sameword@inedtext`, we step to the next element of the list associated to the note. Let it be  $r$ .
  - For the word marked by `\sameword`, we calculate how many time it is called in its line. To do it:
    - \* We get the absolute line number of the current `\sameword`. This absolute line number was stored with list of relative rank for the current `\edtext`. That means, in the previous example, that, if the absolute line number of `\edtext` was 1, that critical notes (A) and (B) were not associated with  $\{1, 2, 2, 3\}$  but with  $\{(1, 1), (2, 1), (2, 1), (3, 1)\}$ . Such method to know the absolute line number associated to a `\sameword` is required because a `\edtext` can be overlap many lines, but `\sameword` can't get it.
    - \* We get the value associated, when reading the auxiliary file, to the pair compose by the current marked word and the current absolute line number. Let this value be  $n$ .



- If  $n > 1$ , that mean the current word appears more than once time in its line.  
In this case, we call `\showwordrank` with the word as first argument and  $r$  as second argument. If the word is called only once, we just print it.

After theory, implementation.

`\get@sw@txt` As the argument of `\sameword` can contain active character if we use `inputenc` with `utf8` option instead of native UTF-8 engine, we store its detokenized content in a macro in order to allow dynamic name of macro with `\csname`.<sup>25</sup>

Because there is a bug with `\detokenize` and  $\text{\LaTeX}$  when using non BMP characters<sup>26</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.

```

1436 \newcommand{\get@sw@txt}[1]{%
1437   \ifxetex%
1438     \xdef\sw@txt{#1}%
1439   \else%
1440     \expandafter\xdef\expandafter\sw@txt\expandafter{\detokenize{#1}}%
1441   \fi%
1442 }%
1443 %

```

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

```

1444 \newcommandx{\sameword}[2][1,usedefault]{%
1445   \leavevmode%
1446   \get@sw@txt{#2}%
1447 %

```

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

```

1448 \unless\ifledRcol%
1449   \csnumgdef{sw@\sw@txt}{\csuse{sw@\sw@txt}+1}%
1450 %

```

Then, write its value to the numbered file.

```

1451 \protected@write\linenum@out{\string\sw@\sw@txt}{\csuse{sw@\sw@txt}
1452 }{#1}}%

```

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

```

1453 \else%
1454   \csnumgdef{sw@\sw@txt@R}{\csuse{sw@\sw@txt@R}+1}%
1455   \protected@write\linenum@outR{\string\sw@\sw@txt}{\csuse{sw@\sw@txt@R}}{#1}}%
1456 \fi%
1457 %

```

<sup>25</sup>See <http://tex.stackexchange.com/q/244538/7712>.

<sup>26</sup><http://sourceforge.net/p/xetex/bugs/108/>

And print the word.

```
1458   #2%
1459 }%
1460 %
```

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

```
\if@addsw61 \newif\if@addsw%
1462 %
```

\@sw The command printed in the auxiliary files.

```
1463 \newcommand{\@sw}[3]{%
1464   \get@sw@txt{#1}%
1465   \unless\ifledRcol%
1466   %
```

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

```
1467   \csxdef{sw@\sw@txt @\the\absline@num @\the\section@num}{#2}%
1468 %
```

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

```
1469   \numdef{\prev@line}{\the\absline@num-1}%
1470   \ifcsundef{sw@\sw@txt @\prev@line @\the\section@num}{%
1471     \csnumgdef{sw@\sw@txt @\prev@line @\the\section@num}{#2-1}%
1472   }{%
1473 %
```

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

```
1474   \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@num}}%
1475 %
```

And do the same thing for the right side.

```
1476   \else%
1477     \csxdef{sw@\sw@txt @\the\absline@numR @\the\section@numR @R}{#2}%
1478     \numdef{\prev@line}{\the\absline@numR-1}%
1479     \ifcsundef{sw@\sw@txt @\prev@line @\the\section@numR @R}{%
1480       \csnumgdef{sw@\sw@txt @\prev@line @\the\section@numR @R}{#2-1}%
1481     }{%
1482       \numdef{\the@sw}{#2-\csuse{sw@\sw@txt @\prev@line @\the\section@numR @R}}%
1483     }%
1484   \fi%
```

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

```

1485 \@tempcnta=\@edtext@level
1486 \@whilenum{\@tempcnta>0}\do{%
1487   \ifcsdef{sw@list@edtext@tmp@\the\@tempcnta}%
1488     {%
1489       \@addswfalse%
1490       \notbool{lemmacommand@\the\@tempcnta}%
1491       {\@addswtrue}%
1492       {\IfStrEq{#3}{inlemma}%
1493        {\@addswtrue}%
1494        {%
1495          \def\do##1{%
1496            \ifnumequal{##1}{\the\@tempcnta}%
1497              {\@addswtrue\listbreak}%
1498              {}}%
1499          }%
1500          \docsvlist{#3}%
1501        }%
1502      }%
1503      \if@addsw%
1504        \letcs{\@tmp}{sw@list@edtext@tmp@\the\@tempcnta}%
1505        \ifledRcol%
1506          \xright@appenditem{\the@sw}{\the\absline@numR}}\to\@tmp%
1507        \else%
1508          \xright@appenditem{\the@sw}{\the\absline@num}}\to\@tmp%
1509        \fi%
1510        \cslet{sw@list@edtext@tmp@\the\@tempcnta}{\@tmp}%
1511      \fi%
1512    }%
1513    {}%
1514    \advance\@tempcnta by -1%
1515  }%
1516 }%
1517 %

```

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

```

1518 \newcommandx{\sameword@inedtext}[2][1,usedefault]{%
1519   \get@sw@txt{#2}%
1520   \unless\ifledRcol%
1521 %

```

Just a precaution.

```

1522   \ifx\sw@list@inedtext\empty%
1523     \def\the@sw{999}%
1524     \def\this@absline{-99}%
1525   \else%
1526 %

```

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

```

1527 \gl@p\sw@list@inedtext\to\@tmp%
1528 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1529 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1530 \fi%
1531 %

```

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

```

1532 \ifcsdef{sw@sw@txt @\this@absline @\the\section@num}{%
1533 \numdef{\prev@line}{\this@absline-1}%
1534 \numdef{sw@atthisline}{\csuse{sw@sw@txt @\this@absline @\the\
section@num}-\csuse{sw@sw@txt @\prev@line @\the\section@num}}%
1535 }%
1536 {\numdef{sw@atthisline}{0}}%
1537 %

```

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

```

1538 \ifnumgreater{sw@atthisline}{1}%
1539 {\showwordrank{#2}{\the@sw}}%
1540 {#2}%
1541 %

```

And the same for right side.

```

1542 \else%
1543 \ifx\sw@list@inedtext\empty%
1544 \def\the@sw{999}%
1545 \def\this@absline{-99}%
1546 \else%
1547 \gl@p\sw@list@inedtext\to\@tmp%
1548 \edef\the@sw{\expandafter\@firstoftwo\@tmp}%
1549 \edef\this@absline{\expandafter\@secondoftwo\@tmp}%
1550 \fi%
1551 \ifcsdef{sw@sw@txt @\this@absline @\the\section@num @R}{%
1552 \numdef{\prev@line}{\this@absline-1}%
1553 \numdef{sw@atthisline}{\csuse{sw@sw@txt @\this@absline @\the\
section@num @R}-\csuse{sw@sw@txt @\prev@line @\the\section@num @R}}%
1554 }%
1555 {\numdef{sw@atthisline}{0}}%
1556 \ifnumgreater{sw@atthisline}{1}%
1557 {\showwordrank{#2}{\the@sw}}%
1558 {#2}%
1559 \fi%
1560 }%
1561 %

```

`\showwordrank`<sup>k62</sup> % Finally, the way the rank will be printed.

```

1563 \newcommand{\showwordrank}[2]{%
1564 #1\textsuperscript{#2}%

```

```

1565 }%
1566 %

```

## 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

`\numberedpar@true` 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

`\numberedpar@false` paragraph when it is complete. When we chop it up into lines, each line in turn goes

`\num@lines` into the `\one@line` register, and `\par@line` will be the number of that line within the

`\one@line` paragraph.

`\par@line`

```

1567 \newbox\raw@text
1568 \newif\ifnumberedpar@
1569 \newcount\num@lines
1570 \newbox\one@line
1571 \newcount\par@line
1572 %

```

`\pstart` `\pstart` starts the paragraph by clearing the `\inserts@list` list and other relevant

`\AtEveryPstart` variables, and then arranges for the subsequent text to go into the `\raw@text` box.

`\numberpstarttrue` `\pstart` needs to appear at the start of every paragraph that is to be numbered; the

`\numberpstartfalse` `\autopar` command below may be used to insert these commands automatically.

`\labelpstarttrue` Beware: everything that occurs between `\pstart` and `\pend` is happening within

`\labelpstartfalse` a group; definitions must be global if you want them to survive past the end of the

`\thepstart` paragraph.

```

1573
1574 \newcommand{\AtEveryPstart}[1]{%
1575   \ifstreempty{#1}%
1576     {\xdef\at@every@pstart{}}%
1577     {\gdef\at@every@pstart{\noindent#1}}%
1578 }%
1579 \xdef\at@every@pstart{}%
1580
1581 \newcounter{pstart}

```

```

1582 \renewcommand{\thepstart}{\bfseries\@arabic\c@pstart}. }
1583 \newif\ifnumberpstart
1584 \numberpstartfalse
1585 \newif\iflabelpstart
1586 \labelpstartfalse
1587 \newcommandx*{\pstart}[1][1]{%
1588   \normal@pars%
1589   \ifstrepty{#1}{\at@every@pstart}{\noindent#1}%
1590   \ifautopar%
1591     \autopar%
1592   \fi%
1593   \ifluatex%
1594     \edef\l@luatextextdir@L{\the\luatextextdir}%
1595   \fi%
1596   \if@nobreak%
1597     \let\@oldnobreak\@nobreaktrue%
1598   \else%
1599     \let\@oldnobreak\@nobreakfalse%
1600   \fi%
1601   \@nobreaktrue%
1602   \ifnumbering \else%
1603     \led@err@PstartNotNumbered%
1604     \beginnumbering%
1605   \fi%
1606   \ifnumberedpar@%
1607     \led@err@PstartInPstart%
1608     \pend%
1609   \fi%
1610   \list@clear{\inserts@list}%
1611   \global\let\next@insert=\empty%
1612   \begingroup\normal@pars%
1613   \global\advance \l@dnumpstartsL\@ne
1614   \global\setbox\raw@text=\vbox\bgroup%
1615     \ifautopar\else%
1616       \ifnumberpstart%
1617         \ifinstanza\else%
1618           \ifsidepstartnum\else%
1619             \thepstart%
1620           \fi%
1621         \fi%
1622       \fi%
1623     \fi%
1624   \numberedpar@true%
1625   \iflabelpstart\protected@edef\@currentlabel%
1626     {\p@pstart\thepstart}
1627   \fi%
1628   \l@dzeropenalties%
1629   \ignorespaces%because not automatically ignored if an optional argument
is used (classical TeX behavior for space after commands)
1630 }

```

```
1631 %
```

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

```
1632 \newcommand*{\pend}[1][1]{\ifnumbering \else%
1633   \led@err@PendNotNumbered%
1634   \fi%
1635   \global\l@dskipversenumberfalse%
1636   \ifnumberedpar@ \else%
1637     \led@err@PendNoPstart%
1638   \fi%
1639 %
```

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.

```
1640 \l@dzeroopenalties%
1641 \endgraf\global\num@lines=\prevgraf\egroup%
1642 \global\par@line=0%
1643 %
```

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.

```
1644 \csnumdef{pstartline}{0}%
1645 \loop\ifvbox\raw@text%
1646   \csnumdef{pstartline}{\pstartline+1}%
1647   \do@line%
1648   \ifbypstart@%
1649     \ifnumequal{\pstartline}{1}{%
1650       \bgroup%
1651       \let\leavevmode\relax%
1652       \setline{1}%
1653       \egroup%
1654       \resetprevline@}{}%
1655     \fi%
1656   \repeat%
1657 %
```

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

```
1658 \flush@notes%
1659 \endgroup%
1660 \ignorespaces%
1661 %
```

Increase pstart counter.

```
1662 \ifnumberpstart%
1663   \pstartnumtrue%
1664 \fi%
1665 \addtocounter{pstart}{1}%
1666 %
```

Restore paragraph, nobreak setting and autopar setting.

```
1667 \normal@pars%
1668 \@oldnobreak%
1669 \ifautopar%
1670   \autopar%
1671 \fi%
1672 %
```

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

```
1673 \ifstrempy{#1}{\at@every@pend}{\noindent#1}%
1674 }
1675
1676 %
```

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.

```
\AtEveryPend77
\at@every@pend78 \newcommand{\AtEveryPend}[1]{%
1679   \ifstrempy{#1}%
1680     {\xdef\at@every@pend{}}%
1681     {\gdef\at@every@pend{\noindent#1}}%
1682 }%
1683 \xdef\at@every@pend{}%
1684
1685 %
```

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

```
1686 \newcommand*\l@dzeroopenalties{%
1687   \brokenpenalty \z@ \clubpenalty \z@
1688   \displaywidowpenalty \z@ \interlinepenalty \z@ \predisplaypenalty \z@
1689   \postdisplaypenalty \z@ \widowpenalty \z@}
1690
1691 %
```

`\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.

```

1692 \newif\ifautopar
1693 \autoparfalse
1694 \newcommand*{\autopar}{
1695   \ifledRcol
1696     \ifnumberingR \else
1697       \led@err@AutoparNotNumbered
1698       \beginnumberingR
1699       \fi
1700     \else
1701       \ifnumbering \else
1702         \led@err@AutoparNotNumbered
1703         \beginnumbering
1704         \fi
1705       \fi
1706       \autopartrue
1707       \everypar{\setbox0=\lastbox
1708         \endgraf \vskip-\parskip
1709         \pstart \noindent \kern\wd0 \ifnumberpstart\ifinstanza\else\thepstart\fi\fi
1710         \let\par=\pend}%
1711       \ignorespaces}
1712 %

```

**\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.

```

1713 \newcommand*{\normal@pars}{\everypar{}\let\par\endgraf}

```

```
1714
1715 %
```

`\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.

```
1716 \newif\ifautopar@pause
1717 %
```

## 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`

```
1718 \newcommand*{\l@dunhbox@line}[1]{\unhbox #1}
1719 \newcommand*{\do@line}{%
1720   {\vbadness=10000
1721     \splittopskip=\z@
1722     \do@linehook
1723   \l@demptyd@ta
1724     \global\setbox\one@line=\vsplit\raw@text to\baselineskip}%
1725   \unvbox\one@line \global\setbox\one@line=\lastbox
1726   \getline@num
1727   \IfStrEq{\led@pb@setting}{before}{\led@check@pb\led@check@nopb}{%}
1728   \ifnum\@lock>\@ne
1729     \inserthangingsymboltrue
1730   \else
1731     \inserthangingsymbolfalse
1732   \fi
1733   \check@pb@in@verse
1734   \ifl@dhidnumber%
1735     \global\l@dhidnumberfalse%
1736     \f@x@l@cks%
1737   \else%
1738     \affixline@num%
1739   \fi%
1740 %
```

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

```
1741 \xifinlist{\the\l@dnumpsstartsL}{\eled@sections@}%
1742   {\print@eledsection}%
1743   {\print@line}%
1744   \IfStrEq{\led@pb@setting}{after}{\led@check@pb\led@check@nopb}{%}
1745 }%
1746 %
```

### VII.2.2 Process for “normal” line

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

```
1747 \def\print@line{
1748 %
```

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

```
1749 \affixpstart@num%
1750 %
```

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

```
1751 \hb@xt@ \linewidth{%
1752 %
```

User hook.

```
1753 \do@insidelinehook%
1754 %
```

Left line number

```
1755 \l@dld@ta%
1756 %
```

Restore marginal and footnotes.

```
1757 \add@inserts\affixside@note%
1758 %
```

Print left notes.

```
1759 \l@dlsn@te
1760 %
```

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

```
1761 {\ledllfill\hb@xt@ \wd\one@line{\new@line%
1762 %
```

If we use Lua<sup>La</sup>T<sub>E</sub>X then restore the direction.

```
1763 \ifluatex%
1764 \luatextextdir\l@luatextextdir@L%
1765 \fi%
1766 %
```

Insert, if needed, the hanging symbol.

```
1767 \inserthangingsymbol %Space kept for backward compatibility
1768 %
```

And so, print the line.

```
1769 \l@dunhbox@line{\one@line}}%
1770 %
```

Right line number

```

1771 \ledrlfill\l@drd@ta%
1772 %

```

Print right notes.

```

1773 \l@drsn@te
1774 }}%
1775 %

```

And reinsert penalties (for page breaking)...

```

1776 \add@penalties%
1777 }
1778 %

```

### 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.

```

1779 \def\print@eledsection{%
1780   \add@inserts\affixside@note%
1781   \numdef{\temp@}{\l@dnumstartsL-1}%
1782   \xifinlist{\temp@}{\eled@sections@}{\@nobreaktrue}{\@nobreakfalse}%
1783   \@eled@sectioningtrue%
1784   \csuse{eled@sectioning@the\l@dnumstartsL}%
1785   \@eled@sectioningfalse%
1786   \global\csundef{eled@sectioning@the\l@dnumstartsL}%
1787   \if@RTL%
1788     \hspace{-3\paperwidth}%
1789     {\hbox{\l@dunhbox@line{\one@line}} \new@line}%
1790   \else%
1791     \hspace{3\paperwidth}%
1792     {\new@line \hbox{\l@dunhbox@line{\one@line}}}%
1793   \fi%
1794   \vskip-\baselineskip%
1795 }
1796 %

```

### 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.

```

1797 \newcommand*{\do@linehook}{}
1798 \newcommand*{\do@insidelinehook}{}
1799 %

```

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

```
1800 \newcommand*{\dolinehook}[1]{\gdef\do@linehook{#1}}%
1801 \newcommand*{\doinsidelinehook}[1]{\gdef\do@insidelinehook{#1}}%
1802
1803 %
```

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

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

```
1804 \l@dcsnotetext \newcommand*{\l@emptyd@ta}{%
1805 \l@dcsnotetext@l \gdef\l@dld@ta{}%
1806 \l@dcsnotetext@r \gdef\l@drd@ta{}%
1807 \gdef\l@dcsnotetext@l{}%
1808 \gdef\l@dcsnotetext@r{}%
1809 \gdef\l@dcsnotetext{}%
1810
1811 %
```

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

```
1812 \l@drsn@te \newcommand*{\l@dlsn@te}{%
1813 \hb@xt@ \z@{\hss\box\l@dldp@rbox\kern\ledlsnotesep}}
1814 \newcommand*{\l@drsn@te}{%
1815 \hb@xt@ \z@{\kern\ledrsnotesep\box\l@drp@rbox\hss}}
1816
1817 %
```

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

```
1818 \newcommand*{\ledllfill}{\hfil}
1819 \newcommand*{\ledrlfill}{}
1820
1821 %
```

## 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.

```
1822 \newcommand*{\getline@num}{%
1823 \global\advance\absline@num \@ne%
1824 \do@actions
1825 \do@ballast
```

```

1826 \ifnumberline
1827   \ifsublines@
1828     \ifnum\sub@lock<\tw@
1829       \global\advance\subline@num \@ne
1830     \fi
1831   \else
1832     \ifnum\@lock<\tw@
1833       \global\advance\line@num \@ne
1834       \global\subline@num \z@
1835     \fi
1836   \fi
1837 \fi
1838 }
1839 %

```

**\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. 135).

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

```

1840 \newcount\ballast@count
1841 \newcounter{ballast}
1842   \setcounter{ballast}{0}
1843 %

```

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.

```

1844 \newcommand*{\do@ballast}{\global\ballast@count \z@
1845   \begingroup
1846     \advance\absline@num \@ne
1847     \ifnum\next@actionline=\absline@num
1848       \ifnum\next@action>-1001\relax
1849         \global\advance\ballast@count by -\c@ballast
1850       \fi
1851     \fi
1852   \endgroup}
1853 %

```

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

It may call itself recursively, and to do this efficiently (using  $\TeX$ 's optimization for tail recursion), we define a control-sequence called `\do@actions@next` that is always the last thing that `\do@actions` does. If there could be more actions to process for this line, `\do@actions@next` is set equal to `\do@actions`; otherwise it is just `\relax`.

```

1854 \newcommand*{\do@actions}{%
1855   \global\let\do@actions@next=\relax
1856   \ifnum\absline@num<\next@actionline\else
1857   %

```

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.

```

1858   \ifnum\next@action>-1001
1859     \global\page@num=\next@action
1860     \ifbypage@
1861       \global\line@num=\z@ \global\subline@num=\z@
1862       \resetprevline@
1863     \fi
1864   %

```

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`.)

```

1865   \else
1866     \ifnum\next@action<-4999
1867       \@l@tempcnta=-\next@action
1868       \advance\@l@tempcnta by -5001
1869       \ifsublines@
1870         \global\subline@num=\@l@tempcnta
1871       \else
1872         \global\line@num=\@l@tempcnta
1873       \fi
1874   %

```

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

```

1875   \else
1876     \@l@tempcnta=-\next@action
1877     \advance\@l@tempcnta by -1000
1878     \do@actions@fixedcode
1879   \fi
1880 \fi
1881 %

```

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.

```

1882   \ifx\actionlines@list\empty
1883     \gdef\next@actionline{1000000}%
1884   \else
1885     \gl@p\actionlines@list\to\next@actionline
1886     \gl@p\actions@list\to\next@action
1887     \global\let\do@actions@next=\do@actions
1888   \fi

```

```

1889 \fi
1890 %
    Make the recursive call, if necessary.
1891 \do@actions@next}
1892
1893 %

```

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

```

1894 \newcommand*{\do@actions@fixedcode}{%
1895 \ifcase\@l@dttempcnta
1896 \or% % 1001
    \global\sublines@true
1897 \or% % 1002
    \global\sublines@false
1898 \or% % 1003
    \global\@lock=\@ne
1899 \or% % 1004
    \ifnum\@lock=\tw@
    \global\@lock=\thr@@
    \else
    \global\@lock=\z@
    \fi
1900 \or% % 1005
    \global\sub@lock=\@ne
1901 \or% % 1006
    \ifnum\sub@lock=\tw@
    \global\sub@lock=\thr@@
    \else
    \global\sub@lock=\z@
    \fi
1902 \or% % 1007
    \l@dskipnumbertrue
1903 \or% % 1008
    \l@dskipversenumbertrue%
1904 \or% % 1009
    \l@dhiddenumbertrue
1905 \else
    \led@warn@BadAction
1906 \fi}
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927 %

```

## 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  $\backslash\text{line@num} \leq \backslash\text{firstlinenum}$ , we compare the two directly instead of making these calculations.

We compute, in the scratch counter  $\backslash\text{@l@tempcnta}$ , 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  $\backslash\text{@l@tempcntb}$  for comparison.

First, the case when we are within a sub-line range.

```
1928 \newcommand*{\affixline@num}{%
1929 %
```

No number is attached if  $\backslash\text{ifl@dskipnumber}$  is TRUE (and then it is set to its normal FALSE value). No number is attached if  $\backslash\text{ifnumberline}$  is FALSE (the normal value is TRUE).

```
1930 \ifledgroupnotesL@else
1931 \ifnumberline
1932 \ifl@dskipnumber
1933 \global\l@dskipnumberfalse
1934 \else
1935 \ifsublines@
1936 \l@tempcntb=\subline@num
1937 \ifnum\subline@num>\c@firstsublinenum
1938 \l@tempcnta=\subline@num
1939 \advance\l@tempcnta by-\c@firstsublinenum
1940 \divide\l@tempcnta by\c@sublinenumincrement
1941 \multiply\l@tempcnta by\c@sublinenumincrement
1942 \advance\l@tempcnta by\c@firstsublinenum
1943 \else
1944 \l@tempcnta=\c@firstsublinenum
1945 \fi
1946 %
```

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.

```
1947 \ch@cksub@l@ck
1948 %
```

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

```
1949 \else
1950 \l@tempcntb=\line@num
1951 %
```

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

```

1952 \ifx\linenumberlist\empty
1953 \ifnum\line@num>\c@firstlinenum
1954 \l@dttempcnta=\line@num
1955 \advance\l@dttempcnta by-\c@firstlinenum
1956 \divide\l@dttempcnta by\c@linenumincrement
1957 \multiply\l@dttempcnta by\c@linenumincrement
1958 \advance\l@dttempcnta by\c@firstlinenum
1959 \else
1960 \l@dttempcnta=\c@firstlinenum
1961 \fi
1962 \else
1963 %

```

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

```

1964 \l@dttempcnta=\line@num
1965 \edef\rem@inder{\linenumberlist,\number\line@num,}%
1966 \edef\sc@n@list{\def\noexpand\sc@n@list
1967 ###1,\number\l@dttempcnta,###2|{\def\noexpand\rem@inder
1968 {####2}}}%
1969 \sc@n@list\expandafter\sc@n@list\rem@inder|
1970 \ifx\rem@inder\empty%
1971 \advance\l@dttempcnta\@ne
1972 \fi
1973 %

```

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

```

1974 \ch@ck@l@ck
1975 \fi
1976 %

```

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

```

1977 \ifnum\l@dttempcnta=\l@dttempcntb
1978 \ifl@dskipversenumber\else
1979 %

```

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  $\LaTeX$  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`.

```

1980 \l@drd@ta
1981 \if@twocolumn
1982 \if@firstcolumn
1983 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1984 \else
1985 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1986 \fi
1987 \else
1988 \@l@tempcntb=\line@margin
1989 \ifnum\@l@tempcntb>\@ne
1990 \advance\@l@tempcntb \page@num
1991 \fi
1992 \ifodd\@l@tempcntb
1993 \gdef\l@drd@ta{\rlap{\rightlinenum}}}%
1994 \else
1995 \gdef\l@dld@ta{\llap{\leftlinenum}}}%
1996 \fi
1997 \fi
1998 \fi
1999 %

```

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

```

2000 \f@x@l@cks
2001 \fi
2002 \fi
2003 \fi
2004 }
2005 %
2006

```

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

```

2007 \newcommand*{\ch@cksub@l@ck}{%
2008 \ifcase\sub@lock
2009 \or
2010 \ifnum\sublock@disp=\@ne
2011 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2012 \fi
2013 \or
2014 \ifnum\sublock@disp=\tw@ \else
2015 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2016 \fi
2017 \or
2018 \ifnum\sublock@disp=\z@

```

```

2019 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2020 \fi
2021 \fi}
2022 %

```

Similarly for line numbers.

```

2023 \newcommand*\ch@ck@l@ck}{%
2024 \ifcase\@lock
2025 \or
2026 \ifnum\lock@disp=\@ne
2027 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2028 \fi
2029 \or
2030 \ifnum\lock@disp=\tw@ \else
2031 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2032 \fi
2033 \or
2034 \ifnum\lock@disp=\z@
2035 \@l@tempcntb=\z@ \@l@tempcnta=\@ne
2036 \fi
2037 \fi}
2038 %

```

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

```

2039 \newcommand*\f@x@l@cks}{%
2040 \ifcase\@lock
2041 \or
2042 \global\@lock=\tw@
2043 \or \or
2044 \global\@lock=\z@
2045 \fi
2046 \ifcase\sub@lock
2047 \or
2048 \global\sub@lock=\tw@
2049 \or \or
2050 \global\sub@lock=\z@
2051 \fi}
2052 %
2053 %

```

## 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:

`\affixpstart@num`  
`\pstartnum`

- 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.

```

\leftpstartnum54
\rightpstartnum55 \newif\ifsidepstartnum
\ifsidepstartnum56 \newcommand*{\affixpstart@num}{%
2057     \ifsidepstartnum
2058         \if@twocolumn
2059             \if@firstcolumn
2060                 \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2061             \else
2062                 \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2063             \fi
2064         \else
2065             \l@dtempcntb=\line@margin
2066             \ifnum\l@dtempcntb>\@ne
2067                 \advance\l@dtempcntb \page@num
2068             \fi
2069             \ifodd\l@dtempcntb
2070                 \gdef\l@drd@ta{\rlap{\rightpstartnum}}}%
2071             \else
2072                 \gdef\l@dld@ta{\llap{\leftpstartnum}}}%
2073             \fi
2074         \fi
2075     \fi
2076 }
2077 %
2078
2079 \newif\ifpstartnum
2080 \pstartnumtrue
2081 \newcommand*{\leftpstartnum}{
2082     \ifpstartnum\thepstart
2083     \kern\linenumsep\fi
2084     \global\pstartnumfalse
2085 }
2086 \newcommand*{\rightpstartnum}{
2087     \ifpstartnum
2088     \kern\linenumsep
2089     \thepstart
2090     \fi
2091     \global\pstartnumfalse
2092 }
2093 %
2094

```

## 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.

```
2095 \list@create{\inserts@list}
2096 %
```

`\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`.

```
2097 \newcommand*{\add@inserts}{%
2098   \global\let\add@inserts@next=\relax
2099 %
```

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

```
2100 \ifx\inserts@list\empty \else
2101 %
```

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.

```
2102 \ifx\next@insert\empty
2103   \ifx\insertlines@list\empty
2104     \global\noteschanged@true
2105     \gdef\next@insert{100000}%
2106   \else
2107     \glp\insertlines@list\to\next@insert
2108   \fi
2109 \fi
2110 %
```

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

```

2111 \ifnum\next@insert=\absline@num
2112   \gl@p\inserts@list\to\@insert
2113   \@insert
2114   \global\let\@insert=\undefined
2115   \global\let\next@insert=\empty
2116   \global\let\add@inserts@next=\add@inserts
2117 \fi
2118 \fi
2119 %

```

Make the recursive call, if necessary.

```

2120 \add@inserts@next}
2121
2122 %

```

## XI.2 Penalties

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

In this code, `\num@lines` is the number of lines in the whole paragraph, and `\par@line` is the line we are working on at the moment. The count `\@l@tempcnta` is used to calculate and accumulate the penalty; it is initially set to the value of `\ballast@count`, which has been worked out in `\do@ballast` above (VIII p. 126). Finally, the penalty is checked to see that it does not go below  $-10000$ .

```

2123 \newcommand*{\add@penalties}{\@l@tempcnta=\ballast@count
2124   \ifnum\num@lines>\@ne
2125     \global\advance\par@line \@ne
2126     \ifnum\par@line=\@ne
2127       \advance\@l@tempcnta \clubpenalty
2128     \fi
2129     \@l@tempcntb=\par@line \advance\@l@tempcntb \@ne
2130     \ifnum\@l@tempcntb=\num@lines
2131       \advance\@l@tempcnta \widowpenalty
2132     \fi
2133     \ifnum\par@line<\num@lines
2134       \advance\@l@tempcnta \interlinepenalty
2135     \fi
2136   \fi
2137   \ifnum\@l@tempcnta=\z@
2138     \relax
2139   \else

```

```

2140 \ifnum\@l@dttempcnta>-10000
2141 \penalty\@l@dttempcnta
2142 \else
2143 \penalty -10000
2144 \fi
2145 \fi}
2146
2147 %

```

### 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  $\text{\TeX}$ , 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.

```

2148 \newcommand*{\flush@notes}{%
2149 \xloop
2150 \ifx\inserts@list\empty \else
2151 \glp\inserts@list\to\@insert
2152 \@insert
2153 \global\let\@insert=\undefined
2154 \repeat}
2155
2156 %

```

**\xloop** `\xloop` is a variant of the PLAIN  $\text{\TeX}$  `\loop` macro, useful when it's hard to construct a positive test using the  $\text{\TeX}$  `\if` commands—as in `\flush@notes` above. One says `\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  $\text{\TeX}$  `\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.

```

2157 \def\xloop#1\repeat{%
2158 \def\body{#1\expandafter\body\fi}%
2159 \body}
2160
2161 %

```



## 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.

```
2162 \def\select@lemmafont#1|#2|#3|#4|#5|#6|#7|{\select@lemmafont#7|}
2163 \def\select@lemmafont#1/#2/#3/#4|%
2164   {\fontencoding{#1}\fontfamily{#2}\fontseries{#3}\fontshape{#4}%
2165   \selectfont}
2166
2167 %
```

### 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.

```
2168 \newcommand*{\footnoteoptions@}[3][1=L,usedefault]{%
2169   \def\do##1{%
2170     \ifstrequal{#1}{L}{% In Leftside
2171       \xright@appenditem{\global\noexpand\settogle{##1@}{#3}}\to\
2172       inserts@list% Switch toogle, in all case
2173       \global\advance\insert@count \@ne% Increment the left insert
2174       counter.
2175     }%
2176     \xright@appenditem{\global\noexpand\settogle{##1@}{#3}}\to\
2177     inserts@listR% Switch toogle, in all case
2178     \global\advance\insert@countR \@ne% Increment the right insert
2179     counter insert.
2180   }%
2181   \notblank{#2}{\docsvlist{#2}}{}% Parsing all options
2182 }
```

### XII.3 Notes language

`\footnotelang@lua` `\footnotelang@lua` is called to remember the information about the direction of a lemma when Lua<sup>2</sup>TeX is used.

```

2182 \newcommand*{\footnotelang@lua}[1][1=L,usedefault]{%
2183   \ifstrequal{#1}{L}{%
2184     \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\luatextextdir
2185     }}\to\inserts@list%Know the dir of lemma
2186     \global\advance\insert@count \@ne%
2187     \xright@appenditem{\csxdef{footnote@luatexpardir}{\the\luatexpardir
2188     }}\to\inserts@list%Know the dir of lemma
2189     \global\advance\insert@count \@ne%
2190     }%
2191     {\%
2192     \xright@appenditem{\csxdef{footnote@luatextextdir}{\the\luatextextdir
2193     }}\to\inserts@listR%Know the dir of lemma
2194     \global\advance\insert@countR \@ne%
2195     }%
2196   }

```

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

```

2197 \newcommand*{\footnotelang@poly}[1][1=L,usedefault]{%
2198   \ifstrequal{#1}{L}{%
2199     \if@RTL%
2200       \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}\to\
2201       inserts@list%Know the language used in the lemma
2202       \global\advance\insert@count \@ne%
2203     \else
2204       \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}\to\
2205       inserts@list%Know the language of lemma
2206       \global\advance\insert@count \@ne%
2207     \fi%
2208     \xright@appenditem{\csxdef{footnote@lang}{\expandonce\language}}\
2209     to\inserts@list%Know the language of lemma
2210     \global\advance\insert@count \@ne%
2211     }%
2212     {\%
2213     \if@RTL
2214       \xright@appenditem{\csxdef{footnote@dir}{@RTLtrue}}\to\
2215       inserts@listR%Know the language of lemma
2216       \global\advance\insert@countR \@ne%
2217     \else
2218       \xright@appenditem{\csxdef{footnote@dir}{@RTLfalse}}\to\
2219       inserts@listR%Know the language of lemma

```

```

2215 \global\advance\insert@countR \@ne%
2216 \fi
2217 \xright@appenditem{{\csxdef{footnote@lang}{\expandonce\language}}}\
to\inserts@listR%Know the language of lemma
2218 \global\advance\insert@countR \@ne%
2219 }%
2220 }
2221 %

```

## 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`.

```

2222 \newcommand*{\footsplitskips}{%
2223 \interlinepenalty=\interfootnotelinepenalty
2224 \unless\ifl@dprintingpages%
2225 \floatingpenalty=\@MM%
2226 \fi%

```

```

2227 \splittopskip=\ht\strutbox \splitmaxdepth=\dp\strutbox
2228 \leftskip=\z@skip \rightskip=\z@skip}
2229
2230 %

```

**\normalfootnoterule** `\normalfootnoterule` is a standard footnote-rule macro, for use by a `footstart` macro: just the same as the PLAIN T<sub>E</sub>X footnote rule.

```

2231 \let\normalfootnoterule=\footnoterule
2232 %

```

## 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.

```

2233 \newcommand{\Xarrangement}[2][1,usedefault]{%
2234   \def\do##1{%
2235     \csname Xarrangement@#2\endcsname{##1}%
2236   }%
2237   \ifstrempy{#1}%
2238     {%
2239       \dolistloop{@series}%
2240     }%
2241     {
2242       \docsvlist{#1}%
2243     }%
2244   }%
2245   %

```

### 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 say 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.

```

2246 \newcommand*{\Xarrangement@normal}[1]{%
2247   \csgdef{series@display#1}{normal}
2248   \expandafter\let\csname #1footstart\endcsname=\normalfootstart
2249   \expandafter\let\csname v#1footnote\endcsname=\normalvfootnote
2250   \expandafter\let\csname #1footfmt\endcsname=\normalfootfmt
2251   \expandafter\let\csname #1footgroup\endcsname=\normalfootgroup
2252   \expandafter\let\csname #1footnoterule\endcsname=%
2253                                     \normalfootnoterule
2254   \count\csname #1footins\endcsname=1000
2255   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2256   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2257   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2258   %

```

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.

```

2259   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2260   %

```

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).

```

2261   \ifnoledgroup@else%
2262     \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2263     \expandafter\let\csname mp#1footgroup\endcsname=\mpnormalfootgroup
2264     \count\csname mp#1footins\endcsname=1000
2265     \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2266     \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2267     \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2268   \fi
2269 }
2270
2271 %

```

`\normalvfootnote` We now begin a series of commands that do ‘normal’ footnote formatting: a format much like that implemented in `PLAIN  $\TeX$` , in which each footnote is a separate paragraph.

`\normalvfootnote` takes the series letter as `#1`, and the entire text of the footnote is `#2`. It does the `\insert` for this note, calling on the `\footfmt` macro for this note series to format the text of the note.

```

2272 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnote}[2]{%
2273   \insert\csname #1footins\endcsname\bgroup
2274   \noindent\csuse{Xbhooknote@#1}%
2275   \csuse{Xnotefontsize@#1}%
2276   \footplitskips
2277   \ifl@dpairing\ifl@dpadding\else%
2278     \setXnoteswidthliketwocolumns@{#1}%
2279   \fi\fi%
2280   \setXnotespositionliketwocolumns@{#1}%
2281   \spaceskip=\z@skip \xspaceskip=\z@skip
2282   \csname #1footfmt\endcsname #2{#1}\egroup}
2283 %

```

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

```

2284 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\mpnormalvfootnote}[2]{%
2285   \global\setbox\@nameuse{mp#1footins}\vbox{%
2286     \unvbox\@nameuse{mp#1footins}
2287     \noindent\csuse{Xbhooknote@#1}%
2288     \csuse{Xnotefontsize@#1}%
2289     \hsize\columnwidth
2290     \@parboxrestore
2291     \color@begingroup
2292     \csname #1footfmt\endcsname #2{#1}\color@endgroup}}
2293 %
2294 %

```

`\normalfootfmt` `\normalfootfmt` is a ‘normal’ macro to take the footnote line and page number information (see V.9 p. 81), 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.

```

2295
2296
2297 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmt}[4]{%
2298   \Xledsetnormalparstuff{#4}%
2299   \hangindent=\csuse{Xhangindent@#4}
2300   \strut{\printlinefootnote{#1}{#4}}%
2301   {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafnt#1|#2}{#2}}%
2302   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2303     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2304     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}\relax%
2305   }}%
2306   #3\strut\par}
2307 %

```

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

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

But if the skip `\preXnotes@` is greater than 0 pt, it is used instead of `\skip\footins` for the first printed series in one page.

The `\leftskip` and `\rightskip` values are both zeroed here. Similarly, these skips are cancelled in the `\vfootnote` macros for the various types of notes. Strictly speaking, this is necessary only if you are using paragraphed footnotes, but we have put it here and in the other `\vfootnote` macros too so that the behavior of `reledmac` in this respect is general across all footnote types. What this means is that any `\leftskip` and `\rightskip` you specify applies to the main text, but not the footnotes. The footnotes continue to be of width `\hsize`.

```
2308 \newcommand*{\normalfootstart}[1]{%
2309 %
```

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

```
2310 \ifdimequal{0pt}{\preXnotes@}{}%
2311 {%
2312 \iftoggle{preXnotes@}{%
2313 \togglefalse{preXnotes@}%
2314 \skip\csname #1footins\endcsname=%
2315 \dimexpr\csuse{preXnotes@}+\csuse{Xaftererrule@#1}\relax%
2316 }%
2317 }%
2318 }%
2319 \vskip\skip\csname #1footins\endcsname%
2320 %
```

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. 182 for the general description of the problem.

```
2321 \leftskip0pt \rightskip0pt
2322 \ifl@dpairing\else%
2323 \hsize=\old@hsize%
2324 \fi%
2325 \setXnoteswidthliketwocolumns@{#1}%
2326 \setXnotespositionliketwocolumns@{#1}%
2327 %
```

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

```

2328 \print@Xfootnoterule{#1}%
2329 \noindent\leavevmode}
2330 %

```

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

```

2331 \newcommand*\normalfootgroup}[1]{%
2332 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}%
2333 \unvbox\csname #1footins\endcsname%
2334 \hsize=\old@hsize%
2335 }%
2336
2337 %

```

**\mpnormalfootgroup** A somewhat different version for minipages. Notes that, in this case, we don not make distinction between \Xfootgroup and \Xfootstarts macro.

```

2338 \unless\ifnoledgroup@
2339 \newcommand*\mpnormalfootgroup}[1]{%
2340 \vskip\skip\@nameuse{mp#1footins}
2341 \ifl@dpairing\ifparledgroup%
2342 \leavevmode\marks\parledgroup@{begin}%
2343 \marks\parledgroup@series{#1}%
2344 \marks\parledgroup@type{Xfootnote}%
2345 \fi\fi\normalcolor%
2346 \ifparledgroup%
2347 \ifl@dpairing%
2348 \else%
2349 \setXnoteswidthliketwocolumns@{#1}%
2350 \setXnotespositionliketwocolumns@{#1}%
2351 \print@Xfootnoterule{#1}%
2352 \fi%
2353 \else%
2354 \setXnoteswidthliketwocolumns@{#1}%
2355 \setXnotespositionliketwocolumns@{#1}%
2356 \print@Xfootnoterule{#1}%
2357 \fi%
2358 \setlength{\parindent}{Opt}
2359 {\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}
2360 \unvbox\csname mp#1footins\endcsname}}
2361 \fi
2362 %

```

### 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  $\text{\TeX}$  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.

```

2363 \newcommand*{\Xarrangement@paragraph}[1]{%
2364   \csgdef{series@display#1}{paragraph}
2365   \expandafter\newcount\csname #1prevpage@num\endcsname
2366   \expandafter\let\csname #1footstart\endcsname=\parafootstart
2367   \expandafter\let\csname v#1footnote\endcsname=\paravfootnote
2368   \expandafter\let\csname #1footfmt\endcsname=\parafootfmt
2369   \expandafter\let\csname #1footgroup\endcsname=\parafootgroup
2370   \count\csname #1footins\endcsname=1000
2371   \csxdef{default@#1footins}{1000}%Use this to confine the notes to one
side only
2372   \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}
2373   \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2374   \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2375   \para@footsetup{#1}
2376   %

```

And the extra setup for minipages.

```

2377   \ifnoledgroup@else
2378     \expandafter\let\csname mpv#1footnote\endcsname=\mpparavfootnote
2379     \expandafter\let\csname mp#1footgroup\endcsname=\mpparafootgroup
2380     \count\csname mp#1footins\endcsname=1000
2381     \dimen\csname mp#1footins\endcsname=\csuse{Xmaxhnotes@#1}
2382     \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2383     \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2384   \fi
2385 }
2386 %

```

`\footfudgefiddle` For paragraphed footnotes  $\text{\TeX}$  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.

```

2387 \providecommand{\footfudgefiddle}{64}
2388 %

```

`\para@footsetup` `\footparagraph` calls the `\para@footsetup` macro to calculate a special fudge factor, which is the ratio of the `\baselineskip` to the `\hspace`. 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  $\TeX$  not `\hsize`. Peter Wilson have also included `\footfudgefiddle`.

```

2389 \newcommand*{\para@footsetup}[1]{\csuse{Xnotefontsize@#1}
2390   \setXnoteswidthliketwocolumns@{#1}%
2391   \dimen0=\baselineskip
2392   \multiply\dimen0 by 1024
2393   \divide \dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\
relax
2394   \csxdef{#1footfudgefactor}{%
2395     \expandafter\strip@pt\dimen0 }}%
2396
2397 %

```

`\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.

```

2398 \newcommand*{\parafootstart}[1]{%
2399   \rightskip=0pt \leftskip=0pt \parindent=0pt
2400   \ifdimequal{0pt}{\preXnotes@}{}%
2401   {%
2402     \iftoggle{preXnotes@}{%
2403       \togglefalse{preXnotes@}%
2404       \skip\csname #1footins\endcsname=%
2405       \dimexpr\csuse{preXnotes@}+\csuse{Xafterterrule@#1}\relax%
2406     }%
2407   }%
2408   }%
2409   \vskip\skip\csname #1footins\endcsname%
2410   \setXnoteswidthliketwocolumns@{#1}%
2411   \setXnotespositionliketwocolumns@{#1}%
2412   \print@Xfootnoterule{#1}%%
2413   \noindent\leavevmode}
2414 %

```

`\paravfootnote` `\paravfootnote` is a version of the `\vfootnote` command that is used for paragraphed notes. It gets appended to the `\inserts@list` list by an outer-level footnote command like `\Afootnote`. The first argument is the note series letter; the second is the full text of the printed note itself, including line numbers, lemmata, and footnote text.

The initial model for this insertion is, of course, the `\insert\footins` definition in *The TeXbook*, p.398. There, the footnotes are first collected up in hboxes, and these hboxes are later unpacked and stuck together into a paragraph.

However, Michael Downes has pointed out that because text in `hboxes` gets typeset in restricted horizontal mode, there are some undesirable side-effects if you later want to break such text across lines. In restricted horizontal mode, where  $\TeX$  does not expect to have to break lines, it does not insert certain items like `\discretionary`s. If you later unbox these `hboxes` and stick them together, as the *TeXbook* macros do to make these footnotes, you lose the ability to hyphenate after an explicit hyphen. This can lead to overfull `hboxes` when you would not expect to find them, and to the uninitiated it might be very hard to see why the problem had arisen.<sup>27</sup>

Wayne Sullivan pointed out to us another subtle problem that arises from the same cause:  $\TeX$  also leaves the `\language` whatsit nodes out of the horizontal list.<sup>28</sup> So changes from one language to another will not invoke the proper hyphenation rules in such footnotes. Since critical editions often do deal with several languages, especially in a footnotes, we really ought to get this bit of code right.

To get around these problems, Wayne suggested emendations to the *TeXbook* versions of these macros which are broadly the same as those described by Michael: the central idea (also suggested by Donald Knuth in a letter to Michael) is to avoid collecting the text in an `\hbox` in the first place, but instead to collect it in a `\vbox` whose width is (virtually) infinite. The text is therefore typeset in unrestricted horizontal mode, as a paragraph consisting of a single long line. Later, there is an extra level of unboxing to be done: we have to unpack the `\vbox`, as well as the `hboxes` inside it, but that is not too hard. For details, we refer you to Michael's article, where the issues are clearly explained.<sup>29</sup> Michael's unboxing macro is called `\Xunvxh`: `unvbox`, extract the last line, and `unhbox` it.

Doing things this way has an important consequence: as Michael pointed out, you really can't put an explicit line-break into a note built in a `\vbox` the way we are doing.<sup>30</sup> In other words, be very careful not to say `\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. 143 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.

```

2415 \newcommand*{\paravfootnote}[2]{%
2416   \insert\csname #1footins\endcsname
2417   \bgroup
2418   \csuse{Xnotefontsize@#1}

```

<sup>27</sup>Michael Downes, 'Line Breaking in `\unhboxed` Text', *TUGboat* **11** (1990), pp. 605–612.

<sup>28</sup>See *The TeXbook*, p. 455 (editions after January 1990).

<sup>29</sup>Wayne supplied his own macros to do this, but since they were almost identical to Michael's, Peter Wilson have used the latter's `\Xunvxh` macro since it is publicly documented.

<sup>30</sup>'Line Breaking', p. 610.

```

2419 \footsplitskips
2420 \setbox0=\vbox{\hsize=\maxdimen
2421   \noindent\csuse{Xhooknote@#1}%
2422   \csname #1footfmt\endcsname #2{#1}}%
2423 \setbox0=\hbox{\Xunvxh{0}{#1}}%
2424 \dp0=0pt
2425 \ht0=\csname #1footfudgefactor\endcsname\wd0
2426 %

```

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

```

2427 \if@RTL\noindent \leavevmode\fi\box0%
2428 \penalty0
2429 \egroup}
2430
2431 %

```

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

`\mpparavfootnote` This version is for minipages.

```

2432 \newcommand*\mpparavfootnote}[2]{%
2433   \global\setbox\@nameuse{mp#1footins}\vbox{%
2434     \unvbox\@nameuse{mp#1footins}%
2435     \csuse{Xnotefontsize@#1}
2436     \footsplitskips
2437     \setbox0=\vbox{\hsize=\maxdimen
2438       \noindent\color@begingroup%
2439       \csuse{Xhooknote@#1}%
2440       \csname #1footfmt\endcsname #2{#1}\color@endgroup}%
2441     \setbox0=\hbox{\Xunvxh{0}{#1}}%
2442     \dp0=\z@
2443     \ht0=\csname #1footfudgefactor\endcsname\wd0
2444     \box0
2445     \penalty0
2446   }}
2447
2448 %

```

`\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). `\unvvh` cancels these unwanted paragraph-final items using `\unskip` and `\unpenalty`.

```

2449 \newcommand*\Xunvvh}[2]{%
2450   \setbox0=\vbox{\unvvhbox#1%
2451     \global\setbox1=\lastbox}%
2452   \unhbox1
2453   \unskip           % remove \rightskip,
2454   \unskip           % remove \parfillskip,
2455   \unpenalty        % remove \penalty of 10000,
2456   \hskip\csuse{Xafternote@#2}} % but add the glue to go between the notes
2457
2458 %

```

`\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).

```

2459 \newcommand*\parafootfmt}[4]{%
2460   \Xinsertparafootsep{#4}%
2461   \Xledsetnormalparstuff{#4}%
2462   \printlinefootnote{#1}{#4}%
2463   {\nottoggle{Xlemmadisablefontselection@#4}}{\select@lemmafnt#1|#2}{#2}}%
2464   \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcsemt{
Xlemmaseparator@#4}%
2465     {\hskip\csuse{Xinplaceoflemmaseparator@#4}}}%
2466     {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}}%
2467   }%
2468   #3\penalty-10 }
2469 %

```

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.

```

2470 \newcommand*\parafootgroup}[1]{%
2471   \unvvh\csname #1footins\endcsname
2472   \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}}%

```

```

2473 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2474 \makeboxofhboxes
2475 \setbox0=\hbox{{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2476 \csuse{Xnotefontsize@#1}
2477 \noindent\unhbox0\par%
2478 \global\hsize=\old@hsize%
2479 }%
2480
2481 %

```

`\mpparafootgroup` The minipage version.

```

2482 \newcommand*{\mpparafootgroup}[1]{%
2483 \setXnoteswidthliketwocolumns@{#1}%
2484 \vskip\skip\@nameuse{mp#1footins}
2485 \ifl@dpairing\ifparledgroup%
2486 \leavevmode\marks\parledgroup@{begin}%
2487 \marks\parledgroup@series{#1}%
2488 \marks\parledgroup@type{Xfootnote}%
2489 \fi\fi\normalcolor
2490 \ifparledgroup%
2491 \ifl@dpairing%
2492 \else%
2493 \setXnoteswidthliketwocolumns@{#1}%
2494 \setXnotespositionliketwocolumns@{#1}%
2495 \print@Xfootnoterule{#1}%%
2496 \fi%
2497 \else%
2498 \setXnoteswidthliketwocolumns@{#1}%
2499 \setXnotespositionliketwocolumns@{#1}%
2500 \print@Xfootnoterule{#1}%
2501 \fi%
2502 \unvbox\csname mp#1footins\endcsname
2503 \ifcsstring{Xragged@#1}{L}{\RaggedLeft}{}%
2504 \ifcsstring{Xragged@#1}{R}{\RaggedRight}{}%
2505 \makeboxofhboxes
2506 \setbox0=\hbox{{\csuse{Xnotefontsize@#1}\csuse{Xtxtbeforenotes@#1}}\
unhbox0 \removehboxes}%
2507 \csuse{Xnotefontsize@#1}
2508 \noindent\unhbox0\par}}
2509
2510 %

```

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

```

\makeboxofhboxes:11 \newcommand*{\makeboxofhboxes}{\setbox0=\hbox{}}%
\removehboxes:12 \loop

```

```

2513 \unpenalty
2514 \setbox2=\lastbox
2515 \ifhbox2
2516 \setbox0=\hbox{\box2\unhbox0}%
2517 \repeat}
2518
2519 \newcommand*{\removehboxes}{\setbox0=\lastbox
2520 \ifhbox0{\removehboxes}\unhbox0 \fi}
2521
2522 %

```

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

```

\prevpage@num 23 \newcommand{\Xinsertparafootsep}[1]{%
\Xinsertparafootsep 24 \ifnumequal{\csuse{#1prevpage@num}}{\page@num}%
2525 {\ifcsdef{prevline#1}% Be sur \prevline#1 exists.
2526 {\ifnumequal{\csuse{prevline#1}}{\line@num}%
2527 {\ifcseempty{Xsymlinenum@#1}{\csuse{Xparafootsep@#1}}{}}%
2528 {\csuse{Xparafootsep@#1}}}%
2529 }%
2530 {\csuse{Xparafootsep@#1}}}%
2531 }%
2532 {}%
2533 \global\csname #1prevpage@num\endcsname=\page@num%
2534 }
2535 %

```

## XII.6.4 Columnar footnotes

### Common tools

`\rigidbalance` We will now define macros for three-column notes and two-column notes. Both sets of macros will use `\rigidbalance`, which splits a box (#1) into into a number (#2) of columns, each with a space (#3) between the top baseline and the top of the `\vbox`. The `\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  $\text{\LaTeX}$  `\line` macro has no relationship to the TeX `\line`. The  $\text{\LaTeX}$  equivalent is `\@@line`.

```

2536 \newcount\@k \newdimen\@h
2537 \newcommand*{\rigidbalance}[3]{\setbox0=\box#1 \@k=#2 \@h=#3
2538 \@@line{\splittopskip=\@h \vbadness=\@M \hfilneg
2539 \valign{##\vfil\cr\dosplits}}}%
2540
2541 \newcommand*{\dosplits}{\ifnum\@k>0 \noalign{\hfil}\splitoff

```

```

2542 \global\advance\@k-1\cr\dosplits\fi}
2543
2544 \newcommand*\splitoff{\dimen0=\ht0
2545 \divide\dimen0 by\@k \advance\dimen0 by\@h
2546 \setbox2 \vsplit0 to \dimen0
2547 \unvbox2 }
2548
2549 %

```

### Three columns

```

\Xarrangement@threecol 2550 \newcommand*\Xarrangement@threecol}[1]{%
2551 \csgdef{series@display#1}{threecol}
2552 \expandafter\let\csname v#1footnote\endcsname=\threecolvfootnote
2553 \expandafter\let\csname #1footfmt\endcsname=\threecolfootfmt
2554 \expandafter\let\csname #1footgroup\endcsname=\threecolfootgroup
2555 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2556 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2557 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2558 \threecolfootsetup{#1}
2559 %

```

The additional setup for minipages.

```

2560 \ifnoledgroup@else
2561 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2562 \expandafter\let\csname mp#1footgroup\endcsname=\mpthreecolfootgroup
2563 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2564 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2565 \mpthreecolfootsetup{#1}
2566 \fi
2567 }
2568
2569 %

```

The `\footstart` and `\footnoterule` macros for these notes assume the normal values (XII.6.2 p. 143 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  $\text{\TeX}$  is accumulating material for the page and checking that limit, it does not apply the  $\backslash\text{count}$  scaling.

```

2570 \newcommand*{\threecolfootsetup}[1]{%
2571   \count\csname #1footins\endcsname 333
2572   \csxdef{default@#1footins}{333}%Use this to confine the notes to one
side only
2573   \multiply\dimen\csname #1footins\endcsname \thr@@}
2574   %

```

$\backslash\text{mpthreecolfootsetup}$  The setup for minipages.

```

2575 \newcommand*{\mpthreecolfootsetup}[1]{%
2576   \count\csname mp#1footins\endcsname 333
2577   \multiply\dimen\csname mp#1footins\endcsname \thr@@}
2578   %
2579   %

```

$\backslash\text{threecolvfootnote}$   $\backslash\text{threecolvfootnote}$  is the  $\backslash\text{vfootnote}$  command for three-column notes. The call to  $\backslash\text{Xnotefontsize@}\langle s \rangle$  ensures that the  $\backslash\text{splittopskip}$  and  $\backslash\text{splitmaxdepth}$  take their values from the right  $\backslash\text{strutbox}$ : the one used in a footnotes. Note especially the importance of temporarily reducing the  $\backslash\text{hsize}$  to 0.3 of its normal value. This determines the widths of the individual columns. So if the normal  $\backslash\text{hsize}$  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).

```

2580 \notbool{parapparatus@}\newcommand*{\newcommand}\threecolvfootnote}[2]{%
2581   \insert\csname #1footins\endcsname\bgroup
2582   \csuse{Xnotefontsize@#1}
2583   \footsplitskip
2584   \csname #1footfmt\endcsname #2{#1}\egroup}
2585   %

```

$\backslash\text{threecolfootfmt}$   $\backslash\text{threecolfootfmt}$  is the command that formats one note. The arguments are #1 the line numbers, #2 the lemma and #4 the text of the  $\text{-footnote}$  command #4 optional (for backward compatibility): the series.

```

2586 \notbool{parapparatus@}\newcommand*{\newcommand}\threecolfootfmt}[4]{%
2587   \normal@pars
2588   \hsize \csuse{Xhsizethreecol@#4}
2589   \nottoggle{Xparindent@#4}\parindent=\z@{}
2590   \tolerance=5000
2591   \hangindent=\csuse{Xhangindent@#4}
2592   \leavevmode
2593   \csuse{Xcolalign@#4}%
2594   \strut{\printlinefootnote{#1}{#4}}%
2595   {\nottoggle{Xlemmadisablefontselection@#4}\select@lemmafont#1|#2}{#2}}%

```

```

2596 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2597 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2598 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2599 }}%
2600 #3\strut\par\allowbreak}
2601 %

```

`\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.

```

2602 \newcommand*{\threecolfootgroup}[1]{\csuse{Xnotefontsize@#1}%
2603 \noindent\csuse{Xtxtbeforenotes@#1}\par%
2604 \splittopskip=\ht\strutbox
2605 \expandafter
2606 \rigidbalance\csname #1footins\endcsname \thr@@ \splittopskip}}
2607 %

```

`\mpthreecolfootgroup` The setup for minipages.

```

2608 \newcommand*{\mpthreecolfootgroup}[1]{\{
2609 \vskip\skip\@nameuse{mp#1footins}
2610 \ifl@dpairing\ifparledgroup%
2611 \leavevmode\marks\parledgroup@{begin}%
2612 \marks\parledgroup@series{#1}%
2613 \marks\parledgroup@type{Xfootnote}%
2614 \fi\fi\normalcolor
2615 \ifparledgroup%
2616 \ifl@dpairing%
2617 \else%
2618 \setXnoteswidthliketwocolumns@{#1}%
2619 \setXnotespositionliketwocolumns@{#1}%
2620 \print@Xfootnoterule{#1}%
2621 \fi%
2622 \else%
2623 \setXnoteswidthliketwocolumns@{#1}%
2624 \setXnotespositionliketwocolumns@{#1}%
2625 \print@Xfootnoterule{#1}%
2626 \fi%
2627 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2628 \splittopskip=\ht\strutbox

```

```

2629 \expandafter
2630 \rigidbalance\csname mp#1footins\endcsname \thr@@ \splittopskip}}
2631
2632 %

```

## Two columns

```

\Xarrangement@twocol 2633 \newcommand*\Xarrangement@twocol}[1]{%
2634 \csgdef{series@display#1}{twocol}
2635 \expandafter\let\csname v#1footnote\endcsname=\twocolvfootnote
2636 \expandafter\let\csname #1footfmt\endcsname=\twocolfootfmt
2637 \expandafter\let\csname #1footgroup\endcsname=\twocolfootgroup
2638 \dimen\csname #1footins\endcsname=\csuse{Xmaxhnotes@#1}%
2639 \skip\csname #1footins\endcsname=\csuse{Xbeforenotes@#1}%
2640 \advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@#1}%
2641 \twocolfootsetup{#1}
2642 %

```

The additional setup for minipages.

```

2643 \ifnoledgroup@else
2644 \expandafter\let\csname mpv#1footnote\endcsname=\mpnormalvfootnote
2645 \expandafter\let\csname mp#1footgroup\endcsname=\mptwocolfootgroup
2646 \skip\csname mp#1footins\endcsname=\csuse{Xbeforenotes@#1}%
2647 \advance\skip\csname mp#1footins\endcsname by\csuse{Xafterrule@#1}%
2648 \mptwocolfootsetup{#1}
2649 \fi
2650 }
2651
2652 %

```

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

```

\twocolvfootnote
\twocolfootfmt
\twocolfootgroup
2653 \newcommand*\twocolfootsetup}[1]{%
2654 \count\csname #1footins\endcsname 500
2655 \csxdef{default@#1footins}{500}%Use this to confine the notes to one
side only
2656 \multiply\dimen\csname #1footins\endcsname \tw@}
2657 %

```

```

2658 \notbool{parapparatus@}\newcommand*\newcommand*\twocolvfootnote}[2]{\
insert\csname #1footins\endcsname\bgroup
2659 \csuse{Xnotefontsize@#1}
2660 \footsplitskips
2661 \csname #1footfmt\endcsname #2{#1}\egroup}
2662 %

```

```

2663 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmt}[4]{% 4th
    arg is optional, for backward compatibility
2664 \normal@pars
2665 \hsize \csuse{Xhsizetwocol@#4}
2666 \nottoggle{Xparindent@#4}{\parindent=\z@}{%
2667 \tolerance=5000
2668 \hangindent=\csuse{Xhangindent@#4}
2669 \leavevmode
2670 \csuse{Xcolalign@#4}%
2671 \strut{\printlinefootnote{#1}{#4}}%
2672 {\nottoggle{Xlemmadisablefontselection@#4}{\select@lemmafont#1|#2}{#2}}%
2673 \iftoggle{nosep@}{\hskip\csuse{Xinplaceoflemmaseparator@#4}}{\ifcempty{
Xlemmaseparator@#4}%
2674 {\hskip\csuse{Xinplaceoflemmaseparator@#4}}%
2675 {\nobreak\hskip\csuse{Xbeforelemmaseparator@#4}\csuse{Xlemmaseparator@
#4}\hskip\csuse{Xafterlemmaseparator@#4}%
2676 }}%
2677 #3\strut\par\allowbreak}
2678 %

2679 \newcommand*{\twocolfootgroup}[1]{\csuse{Xnotefontsize@#1}
2680 \noindent\csuse{Xtxtbeforenotes@#1}\par%
2681 \splittopskip=\ht\strutbox
2682 \expandafter
2683 \rigidbalance\csname #1footins\endcsname \tw@ \splittopskip}}
2684
2685 %

```

`\mptwocolfootsetup` The versions for minipages.

`\mptwocolfootgroup`

```

2686 \newcommand*{\mptwocolfootsetup}[1]{%
2687 \count\csname mp#1footins\endcsname 500
2688 \multiply\dimen\csname mp#1footins\endcsname \tw@}
2689 %

2690 \newcommand*{\mptwocolfootgroup}[1]{%
2691 \vskip\skip\@nameuse{mp#1footins}
2692 \ifl@dpairing\ifparledgroup%
2693 \leavevmode\marks\parledgroup@{begin}%
2694 \marks\parledgroup@series{#1}%
2695 \marks\parledgroup@type{Xfootnote}%
2696 \fi\fi\normalcolor
2697 \ifparledgroup%
2698 \ifl@dpairing%
2699 \else%
2700 \setXnoteswidthliketwocolumns@{#1}%
2701 \setXnotespositionliketwocolumns@{#1}%
2702 \print@Xfootnoterule{#1}%
2703 \fi%
2704 \else%

```

```

2705 \setXnoteswidthliketwocolumns@{#1}%
2706 \setXnotespositionliketwocolumns@{#1}%
2707 \print@Xfootnoterule{#1}%
2708 \fi%
2709 {\csuse{Xnotefontsize@#1}\noindent\csuse{Xtxtbeforenotes@#1}}\par
2710 \splittopskip=\ht\strutbox
2711 \expandafter
2712 \rigidbalance\csname mp#1footins\endcsname \tw@ \splittopskip}}
2713
2714 %

```

## XII.7 Critical notes presentation

Here, we define some commons macro which are used in order to print a critical notes, that is a note with 1) line number 2) lemma 3) lemma separator 4) text associated to the lemma.

### XII.7.1 Font tools

`\endashchar` The fonts that are used for printing notes might not have the character mapping we expect: for example, the Computer Modern font that contains old-style numerals does not contain an en-dash or square brackets, and its period and comma are in odd locations. To allow use of the standard footnote macros with such fonts, we use the following macros for certain characters.

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

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

```

2715 \def\endashchar{\textnormal{--}}
2716 \newcommand*{\fullstop}{\textnormal{.}}
2717 \newcommand*{\rbracket}{\textnormal{}}
2718 \csuse{text\csuse{footnote@lang}}{%
2719     \ifluatex%
2720     \ifdefstring{\footnote@luatextextdir}{TRT}{\thinspace[]{\thinspace
2721 }}%
2722     \else%
2723     \thinspace}%
2724     \fi}%
2725 }
2726
2727 %

```

### XII.7.2 Pstart number in footnote

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

```

2728 \newcommand{\printpstart}[0]{%
2729   \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
2730     \ifledRcol%
2731       \thepstartR%
2732     \else%
2733       \thepstartL%
2734     \fi%
2735   }{%
2736     \thepstart%
2737   }%
2738 }
2739 %

```

### 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`.

```

2740 \newcommand{\printlinefootnote}[2]{%
2741   \def\extractline@##1|##2|##3|##4|##5|##6|##7|{##2}%
2742   \def\extractsubline@##1|##2|##3|##4|##5|##6|##7|{##3}%
2743   \def\extractendline@##1|##2|##3|##4|##5|##6|##7|{##5}%
2744   \def\extractendsubline@##1|##2|##3|##4|##5|##6|##7|{##6}%
2745   \iftoggle{Xnumberonlyfirstintwolines@#2}{%
2746     \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1| - \
extractendline@ #1| - \extractendsubline@ #1|}%
2747     }%
2748     {%
2749       \edef\lineinfo@{\extractline@ #1| - \extractsubline@ #1|}%
2750     }%
2751   \iftoggle{nonum@}{%Try if the line number must printed for this specific
not (by default, yes)
2752     \hspace{\csuse{Xinplaceofnumber@#2}}%
2753   }%
2754   {%
2755     \iftoggle{Xnonumber@#2}%Try if the line number must printed (by
default, yes)
2756     {%
2757       \hspace{\csuse{Xinplaceofnumber@#2}}%
2758     }%
2759   }%
2760 }%

```

```

2761      {\iftoggle{Xnumberonlyfirstinline@#2}% If for this series the
line number must be printed only in the first time.
2762      {%
2763      \ifcsdef{prevline#2}%
2764      {%Be sure the \prevline exists.
2765      \ifcsequal{prevline#2}{\lineinfo@}%Try it
2766      {%
2767      \ifcsequal{Xsymlinenum@#2}%
2768      {%
2769      \hspace{\csuse{Xinplaceofnumber@#2}}%
2770      }%
2771      {\hspace{\csuse{Xbeforeasymlinenum@#2}}\csuse{
Xnotenumfont@#2}%
2772      \ifdimequal{\csuse{Xboxsymlinenum@#2}}{0pt}%
2773      {\csuse{Xsymlinenum@#2}}%
2774      {\hbox to \csuse{Xboxsymlinenum@#2}{\csuse{
Xsymlinenum@#2}\hfill}}%
2775      \hspace{\csuse{Xaftersymlinenum@#2}}}%
2776      }%
2777      {%
2778      \printlinefootnotearea{#1}{#2}%
2779      }%
2780      }%
2781      {%
2782      \printlinefootnotearea{#1}{#2}%
2783      }%
2784      }%
2785      {%
2786      \printlinefootnotearea{#1}{#2}%
2787      }%
2788      \csxdef{prevline#2}{\lineinfo@}%
2789      }%
2790      }%
2791      }%
2792      }%
2793      }
2794      %

```

**\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.)

```

2795 \newcommand{\printlinefootnotearea}[2]{%
2796 \printXbeforenumber{#2}%
2797 \csuse{Xnotenumfont@#2}%
2798 \boxfootnotenumbers{#1}{#2}%
2799 \printXafternumber{#2}%
2800 }%
2801 %

```

**\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.

```

2802 \newcommand{\boxfootnotenumbers}[2]{%
2803   \ifdimequal{\csuse{Xboxlinenum@#2}}{0pt}{%
2804     \printlinefootnotenumbers{#1}{#2}%
2805   }%
2806   {%
2807     \hbox to \csuse{Xboxlinenum@#2}%
2808       {%
2809         \IfSubStr{RC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2810         \printlinefootnotenumbers{#1}{#2}%
2811         \IfSubStr{LC}{\csuse{Xboxlinenumalign@#2}}{\hfill}{}%
2812       }%
2813     }%
2814   }%
2815   %

```

**\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.

```

2816 \newcommand{\printlinefootnotenumbers}[2]{%
2817   \xdef\@currentseries{#2}%
2818   \ifboolexpr{%
2819     (togl{Xpstart@#2} and bool{numberpstart})%
2820     or togl{Xpstarteverytime@#2}}%
2821     {\printpstart}{}%
2822     \iftoggle{Xstanza@#2}{%
2823       \ifnumberstanza%
2824         \printstanza%
2825         \csuse{Xstanzaseparator@#2}%
2826       \fi%
2827     }{%
2828       \iftoggle{Xonlypstart@#2}{\printlines#1|}%
2829     }%
2830   %

```

**\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.)

```

2831 \newcommand{\printXbeforenumber}[1]{%
2832   \hspace{\csuse{Xbeforenumber@#1}}%
2833 }%
2834 %

```

**\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



```

2835 \newcommand{\printXafternumber}[1]{%
2836   \iftoggle{Xnonbreakableafternumber@#1}{\nobreak}{}%
2837   \hspace{\csuse{Xafternumber@#1}}%
2838 }%
2839 %

```

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. 81: 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@pnum40 \newif\ifl@d@pnum
\ifl@d@ssub41 \newif\ifl@d@ssub
\ifl@d@elin42 \newif\ifl@d@elin
\ifl@d@esl43 \newif\ifl@d@esl
\ifl@d@dash44 \newif\ifl@d@dash
\ifl@d@Xtwolines2845 \newif\ifl@d@Xtwolines%
\ifl@d@Xmorethantwolines2846 \newif\ifl@d@Xmorethantwolines%
2847 %

```

<code>\l@dparsefootspec</code>	<code>\l@dparsefootspec{&lt;spec&gt;}{&lt;lemma&gt;}{&lt;text&gt;}</code> parses a footnote specification. <code>&lt;lemma&gt;</code>
<code>\l@dp@rsefootspec</code>	and <code>&lt;text&gt;</code> are the lemma and text respectively. <code>&lt;spec&gt;</code> is the line and page num-
<code>\l@dparsestartpage</code>	ber and lemma font specifier in <code>\l@d@nums</code> style format. The real work is done by
<code>\l@dparsestartline</code>	<code>\l@dp@rsefootspec</code> which defines macros holding the numeric values. Just a reminder
<code>\l@dparsestartsub</code>	of the arguments:
<code>\l@dparseendpage</code>	<code>\printlines    #1         #2     #3        #4     #5     #6        #7</code>
<code>\l@dparseendline</code>	<code>\printlines start-page   line   subline   end-page   line   subline   font</code>
<code>\l@dparseendsub</code>	

```

2848 \newcommand*{\l@dp@rsefootsec}[3]{\l@dp@rsefootsec#1|}
2849 \def\l@dp@rsefootsec#1|#2|#3|#4|#5|#6|#7|{%
2850   \gdef\l@dparsedstartpage{#1}%
2851   \gdef\l@dparsedstartline{#2}%
2852   \gdef\l@dparsedstartsub{#3}%
2853   \gdef\l@dparsedendpage{#4}%
2854   \gdef\l@dparsedendline{#5}%
2855   \gdef\l@dparsedendsub{#6}%
2856 }
2857 %

```

Initialise the several number value macros.

```

2858 \def\l@dparsedstartpage{0}%
2859 \def\l@dparsedstartline{0}%
2860 \def\l@dparsedstartsub{0}%
2861 \def\l@dparsedendpage{0}%
2862 \def\l@dparsedendline{0}%
2863 \def\l@dparsedendsub{0}%
2864
2865 %

```

**\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`.

```

2866 \newcommand*{\setprintlines}[6]{%
2867   \l@d@pnumfalse \l@d@dashfalse
2868 %

```

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

```

2869   \ifbypage@
2870     \ifnum#4=#1 \else
2871       \l@d@pnumtrue
2872       \l@d@dashtrue
2873     \fi
2874   \fi
2875 %

```

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.

```

2876   \ifl@d@pnum \l@d@elintrue \else \l@d@elinfalse \fi
2877   \ifnum#2=#5 \else
2878     \l@d@elintrue
2879     \l@d@dashtrue
2880   \fi
2881 %

```

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

```

2882 \l@d@ssubfalse
2883 \ifnum#3=0 \else
2884     \l@d@ssubtrue
2885 \fi
2886 %

```

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.

```

2887 \l@d@eslfalse
2888 \ifnum#6=0 \else
2889     \ifnum#6=#3
2890         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
2891     \else
2892         \l@d@esltrue
2893         \l@d@dashtrue
2894     \fi
2895 \fi%
2896 %

```

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

```

2897 \ifl@d@dash%
2898     \ifboolexpr{togl{fulllines@} or test{\ifcsemt{Xtwolines@}\
2899     {}%
2900     {%
2901     \setistwofollowinglines{#1}{#2}{#4}{#5}%
2902     \ifboolexpr{%
2903         (%
2904             togl {Xtwolinesbutnotmore@\@currentseries}%
2905             and not%
2906             (%
2907                 bool {istwofollowinglines@}%
2908             )%
2909         )%
2910     or%
2911     (%
2912         (not test{\ifnumequal{#1}{#4}})%
2913         and togl{Xtwolinesonlyinsamepage@\@currentseries}%
2914     )%
2915     }%
2916     {}%
2917     {%
2918     \l@d@dashfalse%
2919     \l@d@Xtwolinesttrue%
2920     \l@d@elinfalse%
2921     \l@d@eslfalse%
2922     \ifcsemt{Xmorethantwolines@\@currentseries}%
2923     {}%

```

```

2924         {\ifistwofollowinglines@else%
2925           \l@d@Xmorethantwolines>true%
2926         \fi%
2927       }%
2928     }%
2929   }%
2930 \fi%
2931 %

```

End of \setprintlines.

```

2932 }%
2933 %

```

**\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.

```

2934 \newif\ifistwofollowinglines@%
2935 \newcommand{\setistwofollowinglines}[4]{%
2936   \ifcsdef{lastlinenumberon@#1}%
2937     {\numdef{\tmp}{\csuse{lastlinenumberon@#1}}}%
2938     {\numdef{\tmp}{0}}}%
2939   \istwofollowinglines@false%
2940   \ifnumequal{#4-#2}{1}%
2941     {\istwofollowinglines@true}%
2942   {\ifbypage@%
2943     \ifnumequal{#3-#1}{1}%
2944     {%
2945       \ifnumequal{#2}{\tmp}%
2946       {\ifnumequal{#4}{1}{\istwofollowinglines@true}}}%
2947     }%
2948   }%

```

```

2949     }%
2950     \fi%
2951   }%
2952 }%
2953 %

```

`\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.

```

2954 \def\printlines#1|#2|#3|#4|#5|#6|#7|{%
2955   \begingroup%
2956 %

```

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

```

2957   \ifluatex%
2958     \luatextextdir TLT%
2959   \fi%
2960 %

```

Decide which part of line number components we will print.

```

2961   \setprintlines{#1}{#2}{#3}{#4}{#5}{#6}%
2962 %

```

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.

```

2963   \ifdimequal{\csuse{Xboxstartlinenum@\@currentseries}}{0pt}%
2964     {\bgroup}%
2965     {\leavevmode\hbox to \csuse{Xboxstartlinenum@\@currentseries}\bgroup\
hfill}%
2966   \ifl@d@pnum #1\fullstop\fi
2967   \linenumrep{#2}
2968   \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
2969   \egroup%
2970 %

```

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

```

2971   \ifdimequal{\csuse{Xboxendlinenum@\@currentseries}}{0pt}%
2972     {\bgroup}%
2973     {\hbox to \csuse{Xboxendlinenum@\@currentseries}\bgroup}%
2974   \ifl@d@Xtwolines%
2975     \ifl@d@Xmorethantwolines%
2976       \csuse{Xmorethantwolines@\@currentseries}%
2977     \else%
2978       \csuse{Xtwolines@\@currentseries}%
2979     \fi%
2980   \else%

```

```

2981 \ifl@d@dash \endashchar\fi%
2982 \ifl@d@pnum #4\fullstop\fi%
2983 \ifl@d@elin \linenumrep{#5}\fi%
2984 \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
2985 \fi%
2986 \ifdimequal{\csuse{Xboxendlinenum@}\@currentseries}}{0pt}%
2987 {}%
2988 {\hfill}%Prevent underfull hbox
2989 \egroup%
2990 \endgroup%
2991 }%
2992 %

```

## 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`

```

2993 \providecommand*\multiplefootnotemarker{3sp}
2994 \providecommand*\multfootsep{\textsuperscript{\normalfont,}}
2995
2996 %

```

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

```

2997 \providecommand*\m@mmf@prepare{%
2998 \kern-\multiplefootnotemarker
2999 \kern\multiplefootnotemarker\relax}
3000 %

```

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

```

3001 \providecommand*\m@mmf@check{%
3002 \ifdim\lastkern=\multiplefootnotemarker\relax

```

```

3003 \edef\x@sf{\the\spacefactor}%
3004 \unkern
3005 \multfootsep
3006 \spacefactor\x@sf\relax
3007 \fi}
3008
3009 %

```

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

```

3010 \@ifclassloaded{memoir}{}{%
3011 %

```

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

```

3012 \apptocmd{\@footnotetext}{\m@mmf@prepare}{}{}
3013 %

```

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

```

3014
3015 \patchcmd{\@footnotemark}
3016 {\nobreak}
3017 {\m@mmf@check
3018 \nobreak
3019 }
3020 {}{}
3021 \patchcmd{\@footnotemark}
3022 {\@makefnmark}
3023 {\@makefnmark
3024 \m@mmf@prepare
3025 }
3026 {}{}
3027 %

```

Finished the modifications for the non-memoir case.

```

3028 }
3029
3030 %

```

## 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.

```

3031 \pretocmd{\@footnotetext}{%
3032   \ifnumberedpar@
3033   \edtext{}\l@dbfnote{#1}}%
3034   \else
3035   {}{}
3036 \apptocmd{\@footnotetext}{\fi}{}{}%
3037 %

```

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

```

3038
3039 \newcommand{\l@dbfnote}[1]{%
3040   \ifnumberedpar@
3041   \gdef\@tag{#1\relax}%
3042   \ifledRcol%
3043     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@thefnmark}}%
3044     \to\inserts@listR
3045     \global\advance\insert@countR \@ne%
3046   \else%
3047     \xright@appenditem{\noexpand\vl@dbfnote{\expandonce\@tag}}{\@thefnmark}}%
3048     \to\inserts@list
3049     \global\advance\insert@count \@ne%
3050   \fi
3051   \fi\ignorespaces}
3052
3053 \newcommand{\vl@dbfnote}[2]{%
3054   \def\@thefnmark{#2}%
3055   \@footnotetext{#1}%
3056   }%
3057 %

```

### 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`

```

3058 \newcommand*\prebodyfootmark{%
3059   \leavevmode
3060   \ifhmode
3061     \edef\@xsf{\the\spacefactor}%
3062     \m@mff@check
3063     \nobreak
3064   \fi}
3065 \newcommand*\postbodyfootmark{%

```



```

3066 \m@mmf@prepare
3067 \ifhmode\spacefactor\@x@sf\fi\relax}
3068
3069 %

```

## 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.

```

3070 \newcommandx{\arrangementX}[2][1,usedefault]{%
3071   \def\do##1{%
3072     \csname arrangementX@#2\endcsname{##1}%
3073   }%
3074   \ifstrepty{#1}%
3075     {%
3076       \dolistloop{\@series}%
3077     }%
3078     {
3079       \docsvlist{#1}%
3080     }%
3081   }%
3082   %

```

### 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.

```

3083 \newcommand*{\normal@footnotemarkX}[1]{%
3084   \prebodyfootmark
3085   \@nameuse{bodyfootmark#1}%
3086   \postbodyfootmark}
3087
3088 %

```

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

```

3089 \newcommand*{\normalbodyfootmarkX}[1]{%
3090   \hbox{\textsuperscript{\normalfont\@nameuse{@thefnmark#1}}}}
3091   %

```

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

```

3092 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalvfootnoteX}[2]{%
3093   \insert\@nameuse{footins#1}\bgroup
3094   \noindent\csuse{bhooknoteX@#1}%
3095   \csuse{notefontsizeX@#1}%
3096   \footsplitskips
3097   \ifl@dpairing\ifl@dpaging\else%
3098     \setnoteswidthliketwocolumnsX@{#1}%
3099   \fi\fi%
3100   \setnotesXpositionliketwocolumns@{#1}%
3101   \spaceskip=\z@skip \xspaceskip=\z@skip
3102   \csuse{\csuse{footnote@dir}}\@nameuse{footfmt#1}{#1}{#2}\egroup}
3103
3104 %

```

`\mpnormalvfootnoteX` The minipage version.

```

3105 \newcommand*{\mpnormalvfootnoteX}[2]{%
3106   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3107     \unvbox\@nameuse{mpfootins#1}
3108     \noindent\csuse{bhooknoteX@#1}%
3109     \csuse{notefontsizeX@#1}%
3110     \hsize\columnwidth
3111     \@parboxrestore
3112     \color@begingroup
3113     \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}}
3114
3115 %

```

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

```

3116 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\normalfootfmtX}[2]{%
3117   \ifluatex%
3118     \luatextextdir\footnote@luatextextdir%
3119     \luatexpardir\footnote@luatexpardir%
3120     \par%
3121   \fi%
3122   \protected@edef\@currentlabel{%
3123     \@nameuse{@thefnmark#1}%
3124   }%
3125   \ledsetnormalparstuffX{#1}%
3126   \hangindent=\csuse{hangindentX@#1}%
3127   {{\csuse{notenunfontX@#1}}\@nameuse{footfootmark#1}}\strut%
3128     #2\strut\par}}
3129
3130 %

```

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

```

3131 \newcommand*{\normalfootfootmarkX}[1]{%
3132   \textsuperscript{\@nameuse{@thefnmark#1}}}
3133
3134 %

```

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

```

3135 \newcommand*{\normalfootstartX}[1]{%
3136   \ifdimequal{Opt}{\prenotesX@}{}%
3137   {%
3138     \iftoggle{prenotesX@}{%
3139       \togglefalse{prenotesX@}%
3140       \skip\csname footins#1\endcsname=%
3141       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3142     }%
3143   }%
3144   }%
3145   \vskip\skip\csname footins#1\endcsname%
3146   \leftskip=\z@
3147   \rightskip=\z@
3148   \ifl@dpairing\else%
3149     \hsize=\old@hsize%
3150   \fi%
3151   \setnoteswidthliketwocolumnsX@{#1}%
3152   \setnotesXpositionliketwocolumns@{#1}%
3153   \print@footnoteXrule{#1}%
3154 }%
3155
3156 %

```

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

```

3157 \let\normalfootnoteruleX=\footnoterule
3158
3159 %

```

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

```

3160 \newcommand*{\normalfootgroupX}[1]{%
3161   \unvbox\@nameuse{footins#1}%
3162   \hsize=\old@hsize%
3163 }%
3164
3165 %

```

**\mpnormalfootgroupX** The minipage version.

```

3166 \newcommand*{\mpnormalfootgroupX}[1]{%
3167   \vskip\skip\@nameuse{mpfootins#1}
3168   \ifl@dpairing\ifparledgroup%
3169     \leavevmode\marks\parledgroup@{begin}%
3170     \marks\parledgroup@series{#1}%
3171     \marks\parledgroup@type{footnoteX}%
3172   \fi\fi\normalcolor
3173   \ifparledgroup%
3174     \ifl@dpairing%
3175     \else%
3176       \setnoteswidthliketwocolumnsX@{#1}%
3177       \setnotesXpositionliketwocolumns@{#1}%
3178       \print@footnoteXrule{#1}%
3179     \fi%
3180   \else%
3181     \setnoteswidthliketwocolumnsX@{#1}%
3182     \setnotesXpositionliketwocolumns@{#1}%
3183     \print@footnoteXrule{#1}%
3184   \fi%
3185   \unvbox\@nameuse{mpfootins#1}}
3186
3187 %

```

### **\normalbfnoteX**<sup>88</sup>

```

3189 \newcommand{\normalbfnoteX}[2]{%
3190   \ifnumberedpar@
3191     \ifledRcol%
3192       \ifluatex
3193         \footnotelang@lua[R]%
3194       \fi
3195       \@ifundefined{xpg@main@language}%if polyglossia
3196       {}%
3197       {\footnotelang@poly[R]}%
3198       \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3199       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3200                               \to\inserts@listR
3201       \global\advance\insert@countR \@ne%
3202     \else%
3203       \ifluatex
3204         \footnotelang@lua%
3205       \fi
3206       \@ifundefined{xpg@main@language}%if polyglossia
3207       {}%
3208       {\footnotelang@poly}%
3209       \protected@xdef\thisfootnote{\csuse{thefootnote#1}}%
3210       \xright@appenditem{\noexpand\vbfnoteX{#1}{#2}{\expandonce\
thisfootnote}}%
3211                               \to\inserts@list

```

```

3212 \global\advance\insert@count \@ne%
3213 \fi
3214 \fi\ignorespaces}
3215
3216 %

```

```

\vbfnoteX\newcommand{\vbfnoteX}[3]{%
3218 \@namedef{@thefnmark#1}{#3}%
3219 \@nameuse{regvfootnote#1}{#1}{#2}}
3220
3221 %

```

```

\vnumfootnoteX\newcommand{\vnumfootnoteX}[2]{%
3223 \ifnumberedpar@
3224 \edtext{}{\normalbfnoteX{#1}{#2}}%
3225 \else
3226 \@nameuse{regvfootnote#1}{#1}{#2}%
3227 \fi}
3228
3229 %

```

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

```

3230 \newcommand*{\arrangementX@normal}[1]{%
3231 \csgdef{series@display#1}{normal}
3232 \expandafter\let\csname footstart#1\endcsname=\normalfootstartX
3233 \expandafter\newcount\csname prevpage#1\endcsname
3234 \@namedef{@footnotemark#1}{\normal@footnotemarkX{#1}}
3235 \@namedef{@bodyfootmark#1}{\normalbodyfootmarkX{#1}}
3236 \expandafter\let\csname regvfootnote#1\endcsname=\normalvfootnoteX
3237 \expandafter\let\csname vfootnote#1\endcsname=\vnumfootnoteX
3238 \expandafter\let\csname footfmt#1\endcsname=\normalfootfmtX
3239 \@namedef{footfootmark#1}{\normalfootfootmarkX{#1}}
3240 \expandafter\let\csname footgroup#1\endcsname=\normalfootgroupX
3241 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3242 \count\csname footins#1\endcsname=1000
3243 \csxdef{default@footins#1}{1000}%Use to have note only for one side
3244 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3245 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3246 \advance\skip\csname footins#1\endcsname by\csuse{afterruleX@#1}%
3247 %

```

Additions for minipages.

```

3248 \ifnoledgroup@else%
3249 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3250 \expandafter\let\csname mpfootgroup#1\endcsname=\mpnormalfootgroupX
3251 \count\csname mpfootins#1\endcsname=1000

```

```

3252 \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3253 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3254 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}%
3255 \fi
3256 }
3257
3258 %

```

### 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%
3259 \newcommand*\arrangementX@twocol}[1]{%
3260 \csgdef{series@displayX#1}{twocol}
3261 \expandafter\let\csname regvfootnote#1\endcsname=\twocolvfootnoteX
3262 \expandafter\let\csname footfmt#1\endcsname=\twocolfootfmtX
3263 \expandafter\let\csname footgroup#1\endcsname=\twocolfootgroupX
3264 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3265 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3266 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3267 \twocolfootsetupX{#1}
3268 \ifnoledgroup@else%
3269 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3270 \expandafter\let\csname mpfootgroup#1\endcsname=\mptwocolfootgroupX
3271 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3272 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
3273 \mptwocolfootsetupX{#1}
3274 \fi%
3275 }
3276
3277 %

```

```

\twocolfootsetupX \twocolfootsetupX{<series>}
\mptwocolfootsetupX
3278 \newcommand*\twocolfootsetupX}[1]{%
3279 \count\csname footins#1\endcsname 500
3280 \csxdef{default@footins#1}{500}%Use this to confine the notes to one
side only
3281 \multiply\dimen\csname footins#1\endcsname by \tw@}
3282 \newcommand*\mptwocolfootsetupX}[1]{%
3283 \count\csname mpfootins#1\endcsname 500
3284 \multiply\dimen\csname mpfootins#1\endcsname by \tw@}
3285
3286 %

```

```

\twocolvfootnoteX \twocolvfootnoteX{<series>}

```

```

3287 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolvfootnoteX}[2]{%
3288   \insert\csname footins#1\endcsname\bgroup
3289     \csuse{notefontsizeX@#1}
3290     \footplitskips
3291     \spaceskip=\z@skip \xspaceskip=\z@skip
3292     \@nameuse{footfmt#1}{#1}{#2}\egroup}
3293 %
3294 %

```

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

```

3295 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\twocolfootfmtX}[2]{%
3296   \protected@edef\@currentlabel{%
3297     \@nameuse{thefnmark#1}%
3298   }%
3299   \normal@pars
3300   \hangindent=\csuse{hangindentX@#1}%
3301   \hsize \csuse{hsizeX@#1}
3302   \nottoggle{parindentX@#1}{\parindent=\z@}{}
3303   \tolerance=5000\relax
3304   \leavevmode
3305   \csuse{colalignX@#1}%
3306   {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3307     #2\strut\par}\allowbreak}
3308 %
3309 %

```

`\twocolfootgroupX` `\twocolfootgroupX{<series>}`  
`\mptwocolfootgroupX`

```

3310 \newcommand*{\twocolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3311   \splittopskip=\ht\strutbox
3312   \expandafter
3313   \rigidbalance\csname footins#1\endcsname \tw@ \splittopskip}}
3314 \newcommand*{\mptwocolfootgroupX}[1]{%
3315   \vskip\skip\@nameuse{mpfootins#1}
3316   \ifl@dpairing\ifparledgroup%
3317     \leavevmode\marks\parledgroup@{begin}%
3318     \marks\parledgroup@series{#1}%
3319     \marks\parledgroup@type{footnoteX}%
3320   \fi\fi\normalcolor
3321   \ifparledgroup%
3322     \ifl@dpairing%
3323     \else%
3324       \setnoteswidthliketwocolumnsX@{#1}%
3325       \setnotesXpositionliketwocolumns@{#1}%
3326       \print@footnoteXrule{#1}%
3327     \fi%
3328   \else%
3329     \setnoteswidthliketwocolumnsX@{#1}%
3330     \setnotesXpositionliketwocolumns@{#1}%

```

```

3331 \print@footnoteXrule{#1}%
3332 \fi%
3333 \splittopskip=\ht\strutbox
3334 \expandafter
3335 \rigidbalance\csname mpfootins#1\endcsname \tw@ \splittopskip}}
3336
3337 %

```

### 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 3338 \newcommand*\arrangementX@threecol}[1]{%
3339 \csgdef{series@displayX#1}{threecol}
3340 \expandafter\let\csname regvfootnote#1\endcsname=\threecolvfootnoteX
3341 \expandafter\let\csname footfmt#1\endcsname=\threecolfootfmtX
3342 \expandafter\let\csname footgroup#1\endcsname=\threecolfootgroupX
3343 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}%
3344 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3345 \advance\skip\csname footins#1\endcsname by \csuse{afterruleX@#1}\relax%
3346 \threecolfootsetupX{#1}
3347 \ifnoledgroup@else%
3348 \expandafter\let\csname mpvfootnote#1\endcsname=\mpnormalvfootnoteX
3349 \expandafter\let\csname mpfootgroup#1\endcsname=\mpthreecolfootgroupX
3350 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3351 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterruleX@#1}
3352 \mpthreecolfootsetupX{#1}
3353 \fi%
3354 }
3355
3356 %

```

```

\threecolfootsetupX \threecolfootsetupX{<series>}
\mpthreecolfootsetupX
3357 \newcommand*\threecolfootsetupX}[1]{%
3358 \count\csname footins#1\endcsname 333
3359 \csxdef{default@footins#1}{333}%Use this to confine the notes to one
side only
3360 \multiply\dimen\csname footins#1\endcsname by \thr@@}
3361 \newcommand*\mpthreecolfootsetupX}[1]{%
3362 \count\csname mpfootins#1\endcsname 333
3363 \multiply\dimen\csname mpfootins#1\endcsname by \thr@@}
3364
3365 %

```

```

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

```



```

3366 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolvfootnoteX}[2]{
3367 %
3368 \insert\csname footins#1\endcsname\bgroup
3369 \csuse{notefontsizeX@#1}
3370 \footsplitskips
3371 \@nameuse{footfmt#1}{#1}{#2}\egroup}
3372 %

```

`\threecolfootfmtX` `\threecolfootfmtX{<series>}`

```

3373 \notbool{parapparatus@}{\newcommand*}{\newcommand}{\threecolfootfmtX}[2]{%
3374 \protected@edef\@currentlabel{%
3375 \@nameuse{@thefnmark#1}%
3376 }%
3377 \hangindent=\csuse{hangindentX@#1}%
3378 \normal@pars
3379 \hsize \csuse{hsizethreecolX@#1}
3380 \nottoggle{parindentX@#1}{\parindent=\z@}{ } %
3381 \tolerance=5000\relax
3382 \leavevmode
3383 \csuse{colalignX@#1}%
3384 {\csuse{notenumfontX@#1}\@nameuse{footfootmark#1}\strut%
3385 #2\strut\par}\allowbreak}
3386 %
3387 %

```

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

```

3388 \newcommand*{\threecolfootgroupX}[1]{\csuse{notefontsizeX@#1}
3389 \splittopskip=\ht\strutbox
3390 \expandafter
3391 \rigidbalance\csname footins#1\endcsname \thr@@ \splittopskip}}
3392 \newcommand*{\mpthreecolfootgroupX}[1]{%
3393 \vskip\skip\@nameuse{mpfootins#1}
3394 \ifl@dpairing\ifparledgroup
3395 \leavevmode\marks\parledgroup@{begin}%
3396 \marks\parledgroup@series{#1}%
3397 \marks\parledgroup@type{footnoteX}%
3398 \fi\fi\normalcolor
3399 \ifparledgroup%
3400 \ifl@dpairing%
3401 \else%
3402 \setnoteswidthliketwocolumnsX@{#1}%
3403 \setnotesXpositionliketwocolumns@{#1}%
3404 \print@footnoteXrule{#1}%
3405 \fi%
3406 \else%
3407 \setnoteswidthliketwocolumnsX@{#1}%
3408 \setnotesXpositionliketwocolumns@{#1}%

```

```

3409 \print@footnoterule{#1}%
3410 \fi%
3411 \splittopskip=\ht\strutbox
3412 \expandafter
3413 \rigidbalance\csname mpfootins#1\endcsname \thr@@ \splittopskip}}
3414
3415 %

```

### XIII.4.5 Paragraphed footnotes

The following macros set footnotes as one paragraph.

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

```

3416 \newcommand*{\arrangementX@paragraph}[1]{%
3417 \csgdef{series@displayX#1}{paragraph}%
3418 \expandafter\newcount\csname #1prevpage@num\endcsname
3419 \expandafter\let\csname footstart#1\endcsname=\parafootstartX
3420 \expandafter\let\csname regvfootnote#1\endcsname=\para@vfootnoteX
3421 \expandafter\let\csname footfmt#1\endcsname=\parafootfmtX
3422 \expandafter\let\csname footgroup#1\endcsname=\para@footgroupX
3423 \expandafter\let\csname footnoterule#1\endcsname=\normalfootnoteruleX
3424 \count\csname footins#1\endcsname=1000
3425 \csxdef{default@footins#1}{1000}%Use this to confine the notes to one
side only
3426 \dimen\csname footins#1\endcsname=\csuse{maxhnotesX@#1}
3427 \skip\csname footins#1\endcsname=\csuse{beforenotesX@#1}%
3428 \advance\skip\csname footins#1\endcsname by\csuse{afterterruleX@#1}%
3429 \para@footsetupX{#1}
3430 \ifnoledgroup@else
3431 \expandafter\let\csname mpvfootnote#1\endcsname=\mppara@vfootnoteX
3432 \expandafter\let\csname mpfootgroup#1\endcsname=\mppara@footgroupX
3433 \count\csname mpfootins#1\endcsname=1000
3434 \dimen\csname mpfootins#1\endcsname=\csuse{maxhnotesX@#1}
3435 \skip\csname mpfootins#1\endcsname=\csuse{beforenotesX@#1}%
3436 \advance\skip\csname mpfootins#1\endcsname by\csuse{afterterruleX@#1}%
3437 \fi
3438 }
3439
3440 %

```

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

```

3441 \newcommand*{\para@footsetupX}[1]{\csuse{notefontsizeX@#1}
3442 \setnoteswidthliketwocolumnsX@{#1}%
3443 \dimen0=\baselineskip
3444 \multiply\dimen0 by 1024
3445 \divide\dimen0 by \columnwidth \multiply\dimen0 by \footfudgefiddle\relax
%
3446 \expandafter

```

```

3447 \xdef\csname footfudgefactor#1\endcsname{%
3448   \expandafter\strip@pt\dimen0 }}
3449
3450 %

```

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

```

3451 \newcommand*{\parafootstartX}[1]{%
3452   \ifdimequal{0pt}{\prenotesX@}{}%
3453   {%
3454     \iftoggle{prenotesX@}{%
3455       \togglefalse{prenotesX@}%
3456       \skip\csname footins#1\endcsname=%
3457       \dimexpr\csuse{prenotesX@}+\csuse{afterruleX@#1}\relax%
3458     }%
3459   }%
3460 }%
3461 \vskip\skip\csname footins#1\endcsname%
3462 \leftskip=\z@
3463 \rightskip=\z@
3464 \parindent=\z@
3465 \vskip\skip\@nameuse{footins#1}%
3466 \setnoteswidthliketwocolumnsX@{#1}%
3467 \setnotesXpositionliketwocolumns@{#1}%
3468 \print@footnoteXrule{#1}%
3469 }
3470
3471 %

```

`\para@vfootnoteX` `\para@vfootnoteX{<series>}{<text>}`  
`\mppara@vfootnoteX`

```

3472 \newcommand*{\para@vfootnoteX}[2]{%
3473   \insert\csname footins#1\endcsname
3474   \bgroup
3475     \csuse{notefontsizeX@#1}
3476     \footsplitskips
3477     \setbox0=\vbox{\hsize=\maxdimen
3478       \noindent\csuse{bhooknoteX@#1}%
3479       \@nameuse{footfmt#1}{#1}{#2}}%
3480     \setbox0=\hbox{\unvXH{0}{#1}}%
3481     \dp0=\z@
3482     \ht0=\csname footfudgefactor#1\endcsname\wd0
3483     \box0
3484     \penalty0
3485   \egroup}
3486 \newcommand*{\mppara@vfootnoteX}[2]{%
3487   \global\setbox\@nameuse{mpfootins#1}\vbox{%
3488     \unvbox\@nameuse{mpfootins#1}
3489     \csuse{notefontsizeX@#1}
3490     \footsplitskips

```

```

3491 \setbox0=\vbox{\hsize=\maxdimen
3492 \noindent\color@begingroup%
3493 \csuse{bhooknoteX@#1}%
3494 \@nameuse{footfmt#1}{#1}{#2}\color@endgroup}%
3495 \setbox0=\hbox{\unvxhX{0}{#1}}%
3496 \dp0=\z@
3497 \ht0=\csname footfudgefactor#1\endcsname\wd0
3498 \box0
3499 \penalty0}}
3500
3501 %

```

```

\unvxhX \newcommand*{\unvxhX}[2]{% 2th is optional for retro-compatibility
3502
3503 \setbox0=\vbox{\unvbox#1%
3504 \global\setbox1=\lastbox}%
3505 \unhbox1
3506 \unskip % remove \rightskip,
3507 \unskip % remove \parfillskip,
3508 \unpenalty % remove \penalty of 10000,
3509 \hskip\csuse{afternoteX@#2}} % but add the glue to go between the notes
3510
3511 %

```

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

```

3512 \newcommand*{\parafootfmtX}[2]{%
3513 \protected@edef\@currentlabel{%
3514 \nameuse{thefnmark#1}%
3515 }%
3516 \insertparafootsepX{#1}%
3517 \ledsetnormalparstuffX{#1}%
3518 {\csuse{notenumfontX@#1}%
3519 \csuse{notenumfontX@#1}%
3520 \@nameuse{footfootmark#1}%
3521 \strut%
3522 #2\penalty-10}}
3523
3524 %

```

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

```

3525 \newcommand*{\para@footgroupX}[1]{%
3526 \unvbox\csname footins#1\endcsname
3527 \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3528 \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3529 \makehboxofhboxes
3530 \setbox0=\hbox{\unhbox0 \removehboxes}%
3531 \csuse{notefontsizeX@#1}
3532 \noindent\unhbox0\par}

```

```

3533 \newcommand*{\mppara@footgroupX}[1]{%
3534   \setnoteswidthliketwocolumnsX@{#1}%
3535   \vskip\skip\@nameuse{mpfootins#1}
3536   \ifl@dpairing\ifparledgroup
3537     \leavevmode%
3538     \leavevmode\marks\parledgroup@{begin}%
3539     \marks\parledgroup@series{#1}%
3540     \marks\parledgroup@type{footnoteX}%
3541   \fi\fi\normalcolor
3542   \ifparledgroup%
3543     \ifl@dpairing%
3544     \else%
3545       \setnoteswidthliketwocolumnsX@{#1}%
3546       \setnotesXpositionliketwocolumns@{#1}%
3547       \print@footnoteXrule{#1}%
3548     \fi%
3549   \else%
3550     \setnoteswidthliketwocolumnsX@{#1}%
3551     \setnotesXpositionliketwocolumns@{#1}%
3552     \print@footnoteXrule{#1}%
3553   \fi%
3554   \unvbox\csname mpfootins#1\endcsname
3555   \ifcsstring{raggedX@#1}{L}{\RaggedLeft}{}%
3556   \ifcsstring{raggedX@#1}{R}{\RaggedRight}{}%
3557   \makehboxofhboxes
3558   \setbox0=\hbox{\unhbox0 \removehboxes}%
3559   \csuse{notefontsizeX@#1}
3560   \noindent\unhbox0\par}}
3561
3562 %

```

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

```

\prevpage@num63 \newcommand{\insertparafootsepX}[1]{%
\Xinsertparafootsep64   \ifnumequal{\csuse{prevpage#1@num}}{\page@num}%
3565     {\csuse{parafootsepX@#1}}%
3566     {}%
3567   }
3568 %

```

## XIV Code common to both critical and familiar footnote in normal arrangement

`\par` should always be redefined to `\endgraf` within the format macro (this is what `\normal@pars` does), to override tricky material in the main text to get the lines numbered automatically (as set up by `\autopar`, for example).

In the case of footnote arranged in a “normal” way, we also must set some setting for paragraph indent and text direction when using Lua<sup>1</sup>TeX.

That why we have defined `\ledsetnormalparstuff@common` in order to make this setting for both familiar and critical notes. This command is called by command to make specific setting to critical or familiar footnote.

```

\ledsetnormalparstuff@common% \newcommand*{\ledsetnormalparstuff@common}{%
\Xledsetnormalparstuff% \ifluatex%
\ledsetnormalparstuffX% \luatextextdir\footnote@luatextextdir%
3572 \luatexpardir\footnote@luatexpardir%
3573 \fi%
3574 \csuse{\csuse{footnote@dir}}%
3575 \normal@pars%
3576 \parfillskip \z@ \@plus 1fil}%
3577
3578 \newcommand*{\Xledsetnormalparstuff}[1]{%
3579 \ledsetnormalparstuff@common%
3580 \nottoggle{Xparindent@#1}{\noindent}{}%\noindent and and not \parindent=0
pt to avoid to break the (bad) change made when moving from ledmac to
eledmac
3581 }%
3582
3583 \newcommand*{\ledsetnormalparstuffX}[1]{%
3584 \ledsetnormalparstuff@common%
3585 \nottoggle{parindentX@#1}{\noindent}{}%\noindent and and not \parindent=0
pt to avoid to break the (bad) change made when moving from ledmac to
eledmac
3586 }%
3587 %

```

## 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@`

```

3588
3589 \newdimen\old@hsize%
3590 \AtBeginDocument{\old@hsize=\hsize}%
3591
3592 \newcommand{\setXnoteswidthliketwocolumns@}[1]{%
3593 \global\let\hsize@fornote=\hsize%
3594 \global\old@hsize=\hsize%
3595 \iftoggle{Xnoteswidthliketwocolumns@#1}%

```

```

3596   {%
3597   \csuse{setwidthliketwocolumns@\columns@position}%
3598   \global\let\hsize@fornote=\hsize%
3599   }%
3600   {}%
3601   \let\hsize=\hsize@fornote%
3602   \let\columnwidth=\hsize@fornote%
3603 }%
3604
3605 \newcommand{\setnoteswidthliketwocolumnsX@}[1]{%
3606   \global\let\hsize@fornote=\hsize%
3607   \global\old@hsize=\hsize%
3608   \iftoggle{noteswidthliketwocolumnsX@#1}%
3609   {%
3610     \csuse{setwidthliketwocolumns@\columns@position}%
3611     \global\let\hsize@fornote=\hsize%
3612   }%
3613   {}%
3614   \let\hsize=\hsize@fornote%
3615   \let\columnwidth=\hsize@fornote%
3616 }%
3617
3618 %

```

`\setnotespositionliketwocolumns@` These two commands set the position of the critical / familiar footnotes, depending on the hooks `Xnoteswidthliketwocolumns` and `noteswidthliketwocolumnsX`. They call commands which are defined only in `reledpar`, because this feature has no sense without `reledpar`.

```

3619 \newcommand{\setXnotespositionliketwocolumns@}[1]{%
3620   \iftoggle{Xnoteswidthliketwocolumns@#1}{%
3621     \csuse{setnotespositionliketwocolumns@\columns@position}%
3622   }{}%
3623 }%
3624
3625 \newcommand{\setnotesXpositionliketwocolumns@}[1]{%
3626   \iftoggle{noteswidthliketwocolumnsX@#1}{%
3627     \csuse{setnotespositionliketwocolumns@\columns@position}%
3628   }{}%
3629 }%
3630
3631 %

```

## 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.

`\mpfnpos`

`\@fnpos`

`\@mpfnpos`

```

3632 \def\@fnpos{familiar-critical}
3633 \def\@mpfnpos{critical-familiar}
3634 \newcommand{\fnpos}[1]{\xdef\@fnpos{#1}}
3635 \newcommand{\mpfnpos}[1]{\xdef\@mpfnpos{#1}}
3636 %

```

## 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@Xfootnoterule 37 \newcommand{\print@Xfootnoterule}[1]{%
\print@footnotexrule 38 \vskip-\csuse{Xafterrule@#1}%Because count in \dimen\csuse{#1footins}
3639 \nointerlineskip%
3640 \moveleft-\leftskip\vbox{\csuse{#1footnoterule}}%
3641 \nointerlineskip%
3642 \vskip\csuse{Xafterrule@#1}%
3643 }%
3644
3645 \newcommand{\print@footnotexrule}[1]{%
3646 \vskip-\csuse{afterruleX@#1}%Because count in \dimen\csuse{footins#1}
3647 \nointerlineskip%
3648 \moveleft-\leftskip\vbox{\csuse{footnoterule#1}}%
3649 \nointerlineskip%
3650 \vskip\csuse{afterruleX@#1}%
3651 }%
3652
3653 %

```

## XVIII Specific skip for first series of footnotes

### XVIII.0.1 Overview

`\Xbeforenotes` inserts a specific skip for the first series of notes in a page. As we can't know in advance which series will be the first, we call `\prepare@preXnotes` before inserting any critical notes, in order to prevent page number overlapping.

1. If it is the first note of the current page, it changes the footnote skip for the series to the value specified to `\Xbeforenotes`. It also keeps the series of the note as the first one of the current page.
2. If it is not the first note of the current page:
  - If the current series is printed after the series kept as the first of the current page, then nothing happens.



- If the current series is printed before the series kept as the first of the current page, then it changes the footnote skip of the current series to the value normally used by the series which was marked as the first of the page. It also keeps the current series as the new first one of the current page.

For example, suppose the series order is A,B. We call first a `\Bfootnote` and a `\Afootnote`. The only skips used are, finally, the skip specific to the first series of the page, and the skip for the B series. If we have not called `\Afootnote`, the only skip used is the skip specific to the first series of the page.

That is perfect.

The series skip and the first series of the current page are reset before the footnotes are printed. Then, the `footstart` macros manage the problem of the first series of the page.

After the rule, the space which is defined by `\Xafterrule` does not depend on whether the series is the first one of the page or not. So we use its normal value for each series.

And now, implementation !

### XVIII.0.2 User level command

`\preXnotes@` If user redefines `\preXnotes@`, via `\preXnotes` to a value greater than 0 pt, this skip will be added before first series notes instead of the notes skip.

```
3654 \newtoggle{preXnotes@}
3655 \toggletrue{preXnotes@}
3656 \newcommand{\preXnotes@}{Opt}
3657 \newcommand*{\preXnotes}[1]{\renewcommand{\preXnotes@}{#1}}
3658 %
```

The same, but for familiar footnotes.

```
\preXnotes59 \newtoggle{prenotesX@}
\preXnotes60 \toggletrue{prenotesX@}
3661 \newcommand{\prenotesX@}{Opt}
3662 \newcommand*{\prenotesX}[1]{\renewcommand{\prenotesX@}{#1}}
3663 %
```

### XVIII.0.3 Internal commands

```
firstXseries@64 \gdef\firstXseries@{}
prepare@preXnotes65 \newcommand{\prepare@preXnotes}[1]{%
3666 \ifdimequal{Opt}{\preXnotes@}%
3667 }%
3668 {%
3669 \IfStrEq{\firstXseries@}{}%
3670 \global\skip\csuse{#1footins}=\preXnotes@%
3671 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterrule@
#1}%
```

```

3672 \gdef\firstXseries@{#1}%
3673 }%
3674 {%
3675 \ifseriesbefore{#1}{\firstXseries@}%
3676 {%
3677 \global\skip\csuse{#1footins}=\csuse{Xbeforenotes@\firstXseries@}%
3678 \global\advance\skip\csname #1footins\endcsname by\csuse{Xafterterrule@
#1}%
3679 \gdef\firstXseries@{#1}%
3680 }%
3681 }%
3682 }%
3683 }%
3684 }
3685 %

```

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

```

firstseriesX@86 \gdef\firstseriesX@{}
prepare@prenotesX87 \newcommand{\prepare@prenotesX}[1]{%
3688 \ifdimequal{0pt}{\prenotesX@}%
3689 }%
3690 {%
3691 \IfStrEq{\firstseriesX@}{-}{%
3692 \global\skip\csuse{footins#1}=\prenotesX@%
3693 \global\advance\skip\csname footins#1\endcsname by\csuse{afterterruleX@
#1}%
3694 \gdef\firstseriesX@{#1}%
3695 }%
3696 {%
3697 \ifseriesbefore{#1}{\firstseriesX@}%
3698 {%
3699 \global\skip\csuse{footins#1}=\csuse{beforenotesX@\firstseriesX@}%
3700 \global\advance\skip\csname footins#1\endcsname by\csuse{afterterruleX@
#1}%
3701 \gdef\firstXseries@{#1}%
3702 }%
3703 }%
3704 }%
3705 }%
3706 }
3707 %

```

## XIX Endnotes

First, check the `noend` option.

```

3708 \ifbool{noend@}{-}{%Used instead of \ifnoend@ to prevent expansion problem
3709 %

```

`\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.

```

3710 \newcommand{\l@dend@open}[1]{%
3711   \global\booltrue{l@dend@#1}%
3712   \expandafter\immediate%
3713   \expandafter\openout%
3714   \csname l@d@#1end\endcsname%
3715   =\jobname.#1end\relax%
3716 }%
3717 \newcommand{\l@dend@close}[1]{%
3718   \global\boolfalse{l@dend@#1}%
3719   \expandafter\immediate%
3720   \expandafter\closeout\csname l@d@#1end\endcsname%
3721 }%
3722 %
3723 %

```

`\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.

```

3724 \newcommand{\l@dend@stuff}{%
3725   \def\do##1{%
3726     \ifbool{l@dend@##1}{%
3727       {\l@dend@open{##1}}%
3728       \expandafter\immediate\expandafter\write\csname l@d@##1end\endcsname{
string\l@d@section{\the\section@num}}%
3729     }%
3730     \dolistloop{\@series}%
3731   }%
3732 %
3733 %

```

`\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.

```

3734 \global\newbool{parapparatus@}{\long}\def\endprint#1#2#3#4#5{
3735 \ifXendinsertsep%
3736 \hskip\csuse{Xendafternote@#4}%
3737 \csuse{Xendsep@#4}%
3738 \else%
3739 \iftoggle{Xendparagraph@#4}%
3740 {\global\Xendinsertsep@true}%
3741 {}%
3742 \fi%
3743 \xdef\@currentseries{#4}%
3744 \def\do##1{%
3745 \toggletrue{##1@}%
3746 }%
3747 \notblank{#5}{\docsvlist{#5}}{}%
3748 \csuse{Xendbhooknote@#4}%
3749 \csuse{Xendnotefontsize@#4}%
3750 \ifbool{expr}%
3751 togl {nonum@}%
3752 or togl {Xendnonumber@#4}%
3753 }%
3754 {\hspace{\csuse{Xendinplaceofnumber@#4}}}%
3755 {\printlineendnotearea{#1}{#4}}%
3756 \nottoggle{Xendlemmadisablefontselection@#4}%
3757 {\select@lemmafont#1|#2}%
3758 {#2}%
3759 \ifbool{expr}%
3760 togl {nosep@}%
3761 or test{\ifcsemt{Xendlemmaseparator@#4}}%
3762 }%
3763 {\hskip\csuse{Xendinplaceoflemmaseparator@#4}}%
3764 {\nobreak%
3765 \hskip\csuse{Xendbeforelemmaseparator@#4}%
3766 \csuse{Xendlemmaseparator@#4}%
3767 \hskip\csuse{Xendafterlemmaseparator@#4}%
3768 }%
3769 #3%
3770 \nottoggle{Xendparagraph@#4}{\par}{}%
3771 \def\do##1{%
3772 \togglefalse{##1@}%
3773 }%
3774 \notblank{#5}{\docsvlist{#5}}{}%
3775 }%
3776
3777 \let\l@d@section=\@gobble
3778
3779 %

```

`\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.)

```

3780 \newcommand{\printlineendnotearea}[2]{%
3781   \bgroup%
3782   \csuse{Xendnotenumfont@#2}%
3783   \ifdimequal{\csuse{Xendboxlinenum@#2}}{0pt}%
3784     {\printendlines#1|}%
3785     {\leavevmode%
3786       \hbox to \csuse{Xendboxlinenum@#2}%
3787       {%
3788         \IfSubStr{RC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}}%
3789         \printendlines#1|}%
3790         \IfSubStr{LC}{\csuse{Xendboxlinenumalign@#2}}{\hfill}{}}%
3791       }%
3792   \egroup%
3793   \enspace%
3794 }%
3795 %

```

**\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.

```

3796 \newcommand*\setprintendlines}[6]{%
3797   \l@dpnumfalse \l@ddashfalse
3798   \ifnum#4=#1 \else
3799     \l@dpnumtrue
3800     \l@ddashtrue
3801   \fi
3802 %

```

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.

```

3803 \ifl@dpnum \l@d@elintrue \else \l@d@elinfalse \fi
3804 \ifnum#2=#5 \else
3805   \l@d@elintrue
3806   \l@ddashtrue
3807 \fi
3808 %

```

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

```

3809 \l@d@ssubfalse
3810 \ifnum#3=0 \else
3811     \l@d@ssubtrue
3812 \fi
3813 %

```

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.

```

3814 \l@d@eslfalse
3815 \ifnum#6=0 \else
3816     \ifnum#6=#3
3817         \ifl@d@elin \l@d@esltrue \else \l@d@eslfalse \fi
3818     \else
3819         \l@d@esltrue
3820         \l@d@dashtrue
3821     \fi
3822 \fi%
3823 %

```

```

3824 \ifl@d@dash%
3825 \ifboolexpr{togl{fulllines@} or test{\ifcempty{Xendtwolines@}\
@currentseries}}}%
3826 {}%
3827 {%
3828 \setistwofollowinglines{#1}{#2}{#4}{#5}%
3829 \ifboolexpr{%
3830     (%
3831         togl {Xendtwolinesbutnotmore@\@currentseries}%
3832         and not%
3833         (%
3834             bool {istwofollowinglines@}%
3835         )%
3836     )%
3837     or%
3838     (%
3839         (not test{\ifnumequal{#1}{#4}})%
3840         and togl{Xendtwolinesonlyinsamepage@\@currentseries}%
3841     )%
3842 }%
3843 {}%
3844 {%
3845 \l@d@dashfalse%
3846 \l@d@Xtwolinesttrue%
3847 \l@d@elinfalse%
3848 \l@d@eslfalse%
3849 \ifcempty{Xendmoreethantwolines@\@currentseries}%
3850 {}%
3851 {\ifistwofollowinglines@\else%
3852     \l@d@Xmoreethantwolinesttrue%
3853     \fi%

```

```

3854         }%
3855     }%
3856 }%
3857 \fi%
3858 %

```

End of `\setprintendlines`.

```

3859 }%
3860 %

```

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

```

3861 \def\printendlines#1|#2|#3|#4|#5|#6|#7|{\begingroup
3862   \setprintendlines{#1}{#2}{#3}{#4}{#5}{#6}%
3863 %

```

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.

```

3864 \ifdimequal{\csuse{Xendboxstartlinenum@\@currentseries}}{0pt}%
3865   {\bgroup}%
3866   {\leavevmode\hbox to \csuse{Xendboxstartlinenum@\@currentseries}\bgroup
\hfill}%
3867 \printnpnum{#1}%
3868 \linenumrep{#2}%
3869 \ifl@d@ssub \fullstop \sublinenumrep{#3}\fi
3870 \egroup%
3871 %

```

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

```

3872 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3873   {\bgroup}%
3874   {\hbox to \csuse{Xendboxendlinenum@\@currentseries}\bgroup}%
3875 \ifl@d@Xtwolines%
3876   \ifl@d@Xmorethantwolines%
3877     \csuse{Xendmorethantwolines@\@currentseries}%
3878   \else%
3879     \csuse{Xendtwolines@\@currentseries}%
3880   \fi%
3881 \else%
3882   \ifl@d@dash \endashchar\fi%
3883   \ifl@d@pnum \printnpnum{#4}\fi%
3884   \ifl@d@elin \linenumrep{#5}\fi%
3885   \ifl@d@esl \ifl@d@elin \fullstop\fi \sublinenumrep{#6}\fi%
3886 \fi%
3887 \ifdimequal{\csuse{Xendboxendlinenum@\@currentseries}}{0pt}%
3888   {}%

```

```

3889 {\hfill}%Prevent underfull hbox
3890 \egroup%
3891 \endgroup%
3892 }%
3893
3894 %

```

**\printnpnum** A macro to print a page number in an endnote.

```

3895 \newcommand*{\printnpnum}[1]{p.#1} }
3896
3897 %

```

**\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.

```

3898 \newif\ifXendinsertsep@%
3899 \newcommand*{\doendnotes}[1]{%
3900   \l@dend@close{#1}%
3901   \begingroup
3902     \makeatletter
3903     \expandafter\let\csname #1end\endcsname=\endprint
3904     \input\jobname.#1end%
3905     \global\Xendinsertsep@false%
3906   \endgroup}
3907 %

```

**\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.

```

3908 \newcommand*{\doendnotesbysection}[1]{%
3909   \l@dend@close{#1}%
3910   \global\expandafter\advance\csname #1end@bysection\endcsname by 1%
3911   \begingroup%
3912     \makeatletter%
3913     \def\l@d@section##1{%
3914       \ifnumequal{##1}{\csname #1end@bysection\endcsname}%
3915       {\cslet{#1end}{\endprint}}%
3916       {\cslet{#1end}{\@gobblefive}}%
3917     }%
3918     \input\jobname.#1end%
3919     \global\Xendinsertsep@false%
3920   \endgroup%
3921 }%
3922 %

```



End of section for end notes

```
3923 }%
3924 %
```

## XX Generate series of notes

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

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

```
3925 \newcommand{\newseries}[1]{%
3926   \def\do##1{\newseries@{##1}}%
3927   \docsvlist{#1}
3928 }
3929 %
```

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

```
3930 \newcommand{\@series}{}
3931 %
```

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

```
\newseries@ 3932 \newcommand{\newseries@}[1]{
3933   %
```

### XX.1 Test if series is still existing

```
3934   \xifinlist{#1}{\@series}{\led@warn@SeriesStillExist{#1}}%
3935   {%
3936   %
```

### XX.2 Init specific to reledpar

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

```
3937   \ifdefined\newseries@par%
3938   \newseries@par{#1}%
3939   \fi%
3940   %
```

### XX.3 For critical footnotes

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

```
3941 \unless\ifnocritical@
3942 %
```

#### XX.3.1 Options

```
3943 \newtoggle{Xparindent@#1}
3944 \newtoggle{Xlemmadisablefontselection@#1}
3945 \csgdef{Xhangindent@#1}{Opt}%
3946 \csgdef{Xragged@#1}{}%
3947 \csgdef{Xhsizetwocol@#1}{0.45 \hsize}%
3948 \csgdef{Xhsizethreecol@#1}{.3 \hsize}%
3949 \csgdef{Xcolalign@#1}{\raggedright}%
3950 \csgdef{Xnotenumfont@#1}{\normalfont}%
3951 \csgdef{Xnotefontsize@#1}{\footnotesize}%
3952 \csgdef{Xbhooknote@#1}{}%
3953
3954 \csgdef{Xboxlinenum@#1}{Opt}%
3955 \csgdef{Xboxlinenumalign@#1}{L}%
3956
3957 \csgdef{Xboxstartlinenum@#1}{Opt}%
3958 \csgdef{Xboxendlinenum@#1}{Opt}%
3959
3960 \csgdef{Xboxsymlinenum@#1}{Opt}%
3961 \newtoggle{Xnumberonlyfirstinline@#1}%
3962 \newtoggle{Xnumberonlyfirstintwolines@#1}%
3963 \csgdef{Xtwolines@#1}{}%
3964 \csgdef{Xmorethantwolines@#1}{}%
3965 \newtoggle{Xtwolinesbutnotmore@#1}%
3966 \newtoggle{Xtwolinesonlyinsamepage@#1}%
3967 \newtoggle{Xonlypstart@#1}%
3968 \newtoggle{Xpstarteverytime@#1}%
3969 \newtoggle{Xpstart@#1}%
3970 \newtoggle{Xstanza@#1}%
3971 \csgdef{Xstanzaseparator@#1}{}%
3972 \csgdef{Xsymlinenum@#1}{}%
3973 \newtoggle{Xnonumber@#1}%
3974 \csgdef{Xbeforenumber@#1}{Opt}%
3975 \csgdef{Xafternumber@#1}{0.5em}%
3976 \newtoggle{Xnonbreakableafternumber@#1}%
3977 \csgdef{Xbeforesymlinenum@#1}{\csuse{Xbeforenumber@#1}}%
3978 \csgdef{Xaftersymlinenum@#1}{\csuse{Xafternumber@#1}}%
3979 \csgdef{Xinplaceofnumber@#1}{1em}%
3980 \global\cslet{Xlemmaseparator@#1}{\rbracket}%
3981 \csgdef{Xbeforelemmaseparator@#1}{0em}%
3982 \csgdef{Xafterlemmaseparator@#1}{0.5em}%
```

```

3983 \csgdef{Xinplaceoflemmaseparator@#1}{1em}%
3984 \csgdef{Xbeforenotes@#1}{1.2em \@plus .6em \@minus .6em}
3985 \csgdef{Xafterrule@#1}{0pt}
3986 \csgdef{Xtxtbeforenotes@#1}{ }
3987 \csgdef{Xmaxhnotes@#1}{0.8\vsizex}
3988 \newtoggle{Xnoteswidthliketwocolumns@#1}%
3989 \csgdef{Xparafootsep@#1}{ }%
3990 \csgdef{Xafternote@#1}{1em plus.4em minus.4em}
3991 %

```

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

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

```

3992 \expandafter\newinsert\csname #1footins\endcsname%
3993 \unless\ifnoledgroup%
3994 \expandafter\newinsert\csname mp#1footins\endcsname%
3995 \fi%
3996 %

```

### 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.

```

3997 \global\newcommand{\parapparatus@}{\expandafter\newcommand\expandafter
*}{\expandafter\newcommand}\csname #1footnote\endcsname[2][ ]{%
3998 \ifnum\@edtext@level>0%
3999 \begingroup%
4000 \newcommand{\content}{##2}%
4001 \ifnumberedpar%
4002 \ifledRcol%
4003 \ifluatex%
4004 \footnotelang@lua[R]%
4005 \fi%
4006 \@ifundefined{xpg@main@language}%if polyglossia
4007 {}%
4008 {\footnotelang@poly[R]}%
4009 \footnoteoptions@[R]{##1}{true}%
4010 \xright@appenditem%
4011 \noexpand\prepare@preXnotes{##1}%
4012 \noexpand\prepare@edindex@fornote{\l@d@nums}%
4013 \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.
4014 \noexpand\setcounter{stanzaR}{\the\c@stanzaR}%Save
stanzaR counter for footnote
4015 \noexpand\csuse{v#1footnote}{##1}%
4016 {\l@d@nums}{\expandonce\tag}{\expandonce\content}}
%

```

```

4017         }\to\inserts@listR
4018         \footnoteoptions@{R}{##1}{false}%
4019         \global\advance\insert@countR \@ne%
4020     \else%
4021         \ifluatex%
4022         \footnotelang@lua%
4023         \fi%
4024         \@ifundefined{xpg@main@language}%if polyglossia
4025         {}%
4026         {\footnotelang@poly}%
4027         \footnoteoptions@{##1}{true}%
4028         \xright@appenditem{%
4029             \noexpand\prepare@preXnotes{#1}%
4030             \noexpand\prepare@edindex@fornote{\l@d@nums}%
4031             \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.
4032             \ifl@dpairing%
4033             \noexpand\setcounter{stanzaL}{\the\c@stanzaL}%Save
stanzaR counter for footnote
4034             \fi%
4035             \noexpand\csuse{v#1footnote}{#1}%
4036             {\l@d@nums}{\expandonce\@tag}{\expandonce\content}}%
%
4037         }\to\inserts@list
4038         \global\advance\insert@count \@ne%
4039         \footnoteoptions@{##1}{false}%
4040     \fi
4041     \else
4042         \csuse{v#1footnote}{#1}{\{0|0|0|0|0|0|0\}{#1}}%
4043     \fi%
4044     \endgroup%
4045 \else%
4046     \led@err@FootnoteWithoutEdtext%
4047 \fi%
4048 \ignorespaces%
4049 }
4050 %

```

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

```

4051     \global\csletcs{#1@@footnote}{#1footnote}
4052 %

```

### XX.3.4 Set standard display

```

4053     \Xarrangement@normal{#1}%
4054 %

```

End of for critical footnotes.

```
4055 \fi
4056 %
```

## XX.4 For familiar footnotes

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

```
4057 \unless\ifnofamiliar@
4058 %
```

### XX.4.1 Options

```
4059 \newtoggle{parindentX@#1}
4060 \csgdef{hangindentX@#1}{Opt}%
4061 \csgdef{raggedX@#1}{}%
4062 \csgdef{hsizetwocolX@#1}{0.45 \hsize}%
4063 \csgdef{hsizethreecolX@#1}{.3 \hsize}%
4064 \csgdef{colalignX@#1}{\raggedright}%
4065 \csgdef{notenumfontX@#1}{\normalfont}%
4066 \csgdef{notefontsizeX@#1}{\footnotesize}%
4067 \csgdef{bhooknoteX@#1}{}%
4068 \csgdef{afterruleX@#1}{Opt}
4069 \csgdef{beforenotesX@#1}{1.2em \@plus .6em \@minus .6em}
4070 \csgdef{maxhnotesX@#1}{0.8\vsizex}%
4071 \newtoggle{noteswidthliketwocolumnsX@#1}%
4072 \csgdef{parafootsepX@#1}{}%
4073 \csgdef{afternoteX@#1}{1em plus.4em minus.4em}
4074 % End of for familiar footnotes.
4075 % \subsubsection{Create inserts, needed to add notes in foot}
4076 % As regards inserts, see chapter 15 of the TeXBook by D. Knuth.
4077 % \begin{macrocode}
4078 \expandafter\newinsert\csname footins#1\endcsname%
4079 \unless\ifnoledgroup@%
4080 \expandafter\newinsert\csname mpfootins#1\endcsname%
4081 \fi%
4082 %
```

### 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.

```
4083 \global\expandafter\newcommand\csname footnote#1\endcsname[1]{%
4084 \begingroup%
4085 \prepare@prenotesX{#1}%
4086 \newcommand{\content}{##1}%
4087 \stepcounter{footnote#1}%
4088 }
```

```

4089         \protected@csxdef{@thefnmark#1}{\csuse{thefootnote#1}}%
4090         \nottoggle{nomk@}%Nomk is set to true when using \
footnoteXnomk with \parpackage
4091         {\csuse{@footnotemark#1}}%
4092         {}%
4093         \ifluatex%
4094         \xdef\footnote@luatexttextdir{\the\luatexttextdir}%
4095         \xdef\footnote@luatexpardir{\the\luatexpardir}%
4096         \fi%
4097         \csuse{vfootnote#1}{#1}{\expandonce\content}\m@mmf@prepare%
4098         \endgroup%
4099     }
4100 %

```

Then define the counters.

```

4101     \newcounter{footnote#1}
4102     \global\expandafter\renewcommand\csname thefootnote#1\endcsname{\
arabic{footnote#1}}
4103 %

```

Do not forget to initialize series

```

4104     \arrangementX@normal{#1}%
4105     \fi
4106 %

```

## XX.5 The endnotes

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

```

4107     \unless\ifnoend@
4108 %

```

### XX.5.1 The auxiliary file

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

```

\l@dend@Xtrue
\l@dend@Xfalse
4109     \expandafter\newwrite\csname l@d@#1end\endcsname%
4110     \expandafter\newif\csname ifl@dend@#1\endcsname%
4111 %

```

### 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.

```

4112 \global\expandafter\newcommandx\csname #1endnote\endcsname[2][1,
4113 usedefault]{%
4114     \bgroup%
4115     \newlinechar='40%
4116     \global\@noneed@Footnotetrue%
4117     \newcommand{\content}{##2}%
4118     \expandafter\immediate\expandafter\write\csname l@d@#1end\
endcsname{%
4119         \expandafter\string\csname #1end\endcsname%
4120         {\ifnumberedpar@l@d@nums\fi}%
4121         {\ifnumberedpar@\expandonce\@tag\fi}%
4122         {\expandonce\content}%
4123         {#1}%
4124         {##1}%
4125         \@percentchar%
4126     }%
4127     \egroup%
4128     \ignorespaces%
4129 }%
4130 %

```

\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.

```

4131 \global\cslet{#1end}{\@gobblefive}
4132 %
4133 %

```

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

```

4134 \global\expandafter\newcount\csname #1end@bysection\endcsname%
4135 %

```

### XX.5.3 The options

```

4136 \csgdef{Xendtwolines@#1}{}%
4137 \csgdef{Xendmorethantwolines@#1}{}%
4138 \newtoggle{Xendtwolinesbutnotmore@#1}{}%
4139 \newtoggle{Xendtwolinesonlyinsamepage@#1}{}%
4140 \newtoggle{Xendlemmadisablefontselection@#1}{}%
4141 \csgdef{Xendnotenumfont@#1}{\normalfont}%
4142 \csgdef{Xendnotefontsize@#1}{\footnotesize}%
4143 \csgdef{Xendbhooknote@#1}{}%
4144
4145 \csgdef{Xendboxlinenum@#1}{Opt}%
4146 \csgdef{Xendboxlinenumalign@#1}{L}%

```

```

4147 \csgdef{Xendboxstartlinenum@#1}{Opt}%
4148 \csgdef{Xendboxendlinenum@#1}{Opt}%
4150
4151 \csgdef{Xendlemmaseparator@#1}{}%
4152 \csgdef{Xendbeforelemmaseparator@#1}{0em}%
4153 \csgdef{Xendafterlemmaseparator@#1}{0.5em}%
4154 \csgdef{Xendinplaceoflemmaseparator@#1}{0.5em}%
4155
4156 \newtoggle{Xendparagraph@#1}%
4157 \csgdef{Xendafternote@#1}{1em plus.4em minus.4em}%
4158 \csgdef{Xendsep@#1}{}%
4159
4160 \csgdef{Xendinplaceofnumber@#1}{Opt}%
4161 \newtoggle{Xendnonumber@#1}%
4162 %

```

End of endnotes declaration

```

4163 \fi%
4164 %

```

Dump series in \@series

```

4165 \listxadd{\@series}{#1}
4166 }
4167 }% End of \newseries
4168 %

```

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

```

4169 \expandafter\newseries\expandafter{\default@series}
4170 %

```

# 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.

```

4171 \newcommand{\seriesatbegin}[1]{%
4172 \StrDel{\@series}{#1}[\@series]%
4173 \edef\@new{%
4174 \listadd{\@new}{#1}%
4175 \listadd{\@new}{\@series}%
4176 \xdef\@series{\@new}%
4177 }
4178 %

```

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



```

4179 \newcommand{\seriesatend}[1]{%
4180   \StrDel{\@series}{#1}[\@series]%
4181   \edef\@new{%
4182     \listead{\@new}{\@series}%
4183     \listead{\@new}{#1}%
4184     \xdef\@series{\@new}%
4185   }
4186   %

```

## XXI.2 Test series order

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

```

4187 \newcommand{\ifseriesbefore}[4]{%
4188   \StrPosition{\@series}{#1}[\@first]%
4189   \StrPosition{\@series}{#2}[\@second]%
4190   \ifnumgreater{\@second}{\@first}{#3}{#4}%
4191 }
4192 %

```

### XXI.2.1 Get the first series

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

```

\@getfirstseries93 \newcommand{\@getfirstseries}{%
4194   \ifdefempty{\@series}%
4195   {\xdef\@firstseries{}}%
4196   {\StrChar{\@series}{1}[\@firstseries]}%
4197 }%
4198 %

```

## 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`.

```

4199 \newcommandx{\settoggle@series}[5][4,5,usedefault]{%
4200   \def\do##1{%
4201     \global\settoggle{#2@##1}{#3}%
4202     \ifstrequal{#4}{reload}%
4203       {%
4204         \csuse{Xarrangement@}\csuse{series@display##1}{##1}%
4205         \csuse{arrangementX@}\csuse{series@displayX##1}{##1}%
4206       }%
4207     {%
4208     }%
4209   \ifstreempty{#1}{%
4210     \dolistloop{\@series}%
4211     \ifstreempty{#5}{%
4212       \docsvlist{#5}%
4213     }
4214   }%
4215   {%
4216     \docsvlist{#1}%
4217   }%
4218 }
4219 %

```

`\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`.

```

4220 \newcommandx{\setcommand@series}[5][4,5,usedefault]{%
4221   \def\do##1{
4222     \csgdef{#2@##1}{#3}
4223     \ifstrequal{#4}{reload}{
4224       \csuse{Xarrangement@}\csuse{series@display##1}{##1}%
4225       \csuse{arrangementX@}\csuse{series@displayX##1}{##1}%
4226     }{}
4227     \ifstreempty{#1}{%
4228       \dolistloop{\@series}%
4229       \ifstreempty{#5}{}%
4230       \docsvlist{#5}
4231     }
4232   }%
4233   {%
4234     \docsvlist{#1}%
4235   }%
4236 }%
4237 %

```

### XXI.3.3 Tools to generate options commands

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

```

4238 \newcommandx{\newhookcommand@series}[2][2,usedefault]{%
4239   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4240     \setcommand@series{##1}{#1}{##2}[][#2]%
4241   }%
4242   \ifstreempty{#2}{}%
4243   \def\do##1{%
4244     \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname
4245     [1]{%
4246       \csuse{#1}[##1]{####1}%
4247     }%

```

```

4247 }%
4248 \docsvlist{#2}%
4249 }%
4250 }
4251 %

```

**\newhooktoggle@series** `\newhooktoggle@series\command` names 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><pseudoseris>`, like `\Xtwolinesbutnotmoreappref`.

```

4252 \newcommandx{\newhooktoggle@series}[2][2,usedefault]{%
4253   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
4254     true},usedefault]{%
4255     \settoggle@series{##1}{#1}{##2}[][#2]%
4256   }%
4257   \ifstrempy{#2}{}{%
4258     \def\do##1{%
4259       \global\expandafter\newcommand\expandafter*\csname #1##1\endcsname{%
4260         \csuse{#1}[##1]%
4261       }%
4262     }%
4263     \docsvlist{#2}%
4264   }%
4265 }%

```

**\newhooktoggle@series@reload** `\newhookcommand@toggle@reload` does the same thing as `\newhooktoggle@series` but the commands created by this macro also reload the series arrangement.

```

4266 \newcommand{\newhooktoggle@series@reload}[1]{%
4267   \global\expandafter\newcommandx\expandafter*\csname #1\endcsname[2][1,2={
4268     true},usedefault]{%
4269     \settoggle@series{##1}{#1}{##2}[reload]%
4270   }%
4271 }%

```

**\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.

```

4272 \newcommand{\newhookcommand@series@reload}[1]{%
4273   \global\expandafter\newcommand\expandafter*\csname #1\endcsname[2][]{%
4274     \setcommand@series{##1}{#1}{##2}[reload]%
4275   }%
4276 }%
4277 %

```

### 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.

```

4278 \unless\ifnocritical@
4279   \newhooktoggle@series{Xparindent}
4280   \newhookcommand@series{Xtwolines}[appref]
4281   \newhookcommand@series{Xmorethantwolines}[appref]
4282   \newhooktoggle@series{Xtwolinesbutnotmore}[appref]
4283   \newhooktoggle@series{Xtwolinesonlyinsamepage}[appref]
4284   \newhookcommand@series{Xhangindent}
4285   \newhookcommand@series{Xragged}
4286   \newhookcommand@series{Xhsizetwocol}
4287   \newhookcommand@series{Xhsizethreecol}
4288   \newhookcommand@series{Xcolalign}%
4289   \newhookcommand@series{Xnotenumfont}
4290   \newhookcommand@series{Xbhooknote}
4291   \newhookcommand@series{Xboxsymlinenum}%
4292   \newhookcommand@series{Xsymlinenum}
4293   \newhookcommand@series{Xbeforenumber}
4294   \newhookcommand@series{Xafternumber}
4295   \newhookcommand@series{Xbeforesymlinenum}
4296   \newhookcommand@series{Xaftersymlinenum}
4297   \newhookcommand@series{Xinplaceofnumber}
4298   \newhookcommand@series{Xlemmaseparator}
4299   \newhookcommand@series{Xbeforelemmaseparator}
4300   \newhookcommand@series{Xafterlemmaseparator}
4301   \newhookcommand@series{Xinplaceoflemmaseparator}
4302   \newhookcommand@series{Xtxtbeforenotes}
4303   \newhookcommand@series@reload{Xafterrule}
4304   \newhooktoggle@series{Xnumberonlyfirstinline}
4305   \newhooktoggle@series{Xnumberonlyfirstintwolines}
4306   \newhooktoggle@series{Xnonumber}
4307   \newhooktoggle@series{Xpstart}
4308   \newhooktoggle@series{Xpstarteverytime}%
4309
4310   \newhooktoggle@series{Xstanza}%
4311   \newhookcommand@series{Xstanzaseparator}%
4312
4313   \newhooktoggle@series{Xonlypstart}
4314   \newhooktoggle@series{Xnonbreakableafternumber}
4315   \newhooktoggle@series{Xlemmadisablefontselection}
4316   \newhookcommand@series@reload{Xmaxhnotes}
4317   \newhookcommand@series@reload{Xbeforenotes}
4318   \newhooktoggle@series@reload{Xnoteswidthliketwocolumns}%
4319   \newhookcommand@series{Xnotefontsize}
4320
4321   \newhookcommand@series{Xboxlinenum}%

```

```

4322 \newhookcommand@series{Xboxlinenumalign}%
4323
4324 \newhookcommand@series{Xboxstartlinenum}%
4325 \newhookcommand@series{Xboxendlinenum}%
4326
4327 \newhookcommand@series{Xafternote}%
4328 \newhookcommand@series{Xparafootsep}
4329
4330 \fi
4331 %

```

### XXI.3.5 Options for familiar notes

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

```

4332 \unless\ifnofamiliar@
4333 \newhooktoggle@series{parindentX}
4334 \newhookcommand@series{hangindentX}
4335 \newhookcommand@series{raggedX}
4336 \newhookcommand@series{hsizetwocolX}
4337 \newhookcommand@series{hsizethreecolX}
4338 \newhookcommand@series{colalignX}%
4339 \newhookcommand@series{notenumfontX}
4340 \newhookcommand@series{bhooknoteX}
4341 \newhookcommand@series@reload{beforenotesX}
4342 \newhookcommand@series@reload{maxhnotesX}
4343 \newhooktoggle@series@reload{noteswidthliketwocolumnsX}%
4344 \newhookcommand@series@reload{afterruleX}
4345 \newhookcommand@series{notefontsizeX}
4346 \newhookcommand@series{afternoteX}
4347 \newhookcommand@series{parafootsepX}
4348 \fi
4349 %

```

### 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.

```

4350 \unless\ifnoend@
4351 \newhookcommand@series{Xendtwolines}[apprefwithpage]
4352 \newhookcommand@series{Xendmoreethantwolines}[apprefwithpage]
4353 \newhooktoggle@series{Xendtwolinesbutnotmore}[apprefwithpage]
4354 \newhooktoggle@series{Xendtwolinesonlyinsamepage}[apprefwithpage]
4355 \newhookcommand@series{Xendnotenumfont}
4356 \newhookcommand@series{Xendbhooknote}
4357
4358 \newhookcommand@series{Xendboxlinenum}%

```

```

4359 \newhookcommand@series{Xendboxlinenumalign}%
4360
4361 \newhookcommand@series{Xendboxstartlinenum}%
4362 \newhookcommand@series{Xendboxendlinenum}%
4363
4364 \newhookcommand@series{Xendnotefontsize}
4365 \newhooktoggle@series{Xendlemmadisablefontselection}
4366 \newhookcommand@series{Xendlemmaseparator}
4367 \newhookcommand@series{Xendbeforelemmaseparator}
4368 \newhookcommand@series{Xendafterlemmaseparator}
4369 \newhookcommand@series{Xendinplaceoflemmaseparator}
4370
4371 \newhooktoggle@series{Xendparagraph}
4372 \newhookcommand@series{Xendafternote}
4373 \newhookcommand@series{Xendsep}
4374
4375 \newhookcommand@series{Xendinplaceofnumber}%
4376 \newhooktoggle@series{Xendnonumber}%
4377 \fi
4378 %

```

## XXI.4 Hooks for a particular footnote

**\fulllines@** \fulllines@ toggle is used to print the full lines references, and not the abbreviated form defined by \Xtwolines and \Xmorethantwolines.

```

4379 \newtoggle{fulllines@}%
4380 %

```

**\nonum@** \nonum@ toggle is used to disable line number printing in a particular footnote.

```

4381 \newtoggle{nonum@}
4382 %

```

**\nosep@** \nosep@ toggle is used to disable the lemma separator in a particular footnote.

```

4383 \newtoggle{nosep@}
4384 %

```

**\nomk@** \nomk@ toggle is used by reledpar to remove the footnote mark in the text when using \footnoteXmk. Read reledpar handbook.

```

4385 \newtoggle{nomk@}%
4386 %

```

## XXI.5 Alias

**\Xnolemmaseparator** \Xnolemmaseparator[⟨series⟩] is just an alias for \Xlemmaseparator[⟨series⟩]{}.

```

4387 \newcommand*{\Xnolemmaseparator}[1][1]{\Xlemmaseparator[#1]}
4388 %

```

## 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.

```

4389 \countdef\pageno=0 \pageno=1
4390 \newcommand*{\advancepageno}{\ifnum\pageno<\z@ \global\advance\pageno\m@ne
4391   \else\global\advance\pageno\@ne\fi}
4392
4393 %

```

### XXII.0.2 Extra footnotes output

With luck we might only have to change `\@makecol` and `\@reinserts` of the  $\TeX$ 's kernel. Since `reledmac`, we use `etoolbox`'s patching commands instead of overriding. It should provides better compatibility with other package which modify these commands

`\doextrafeet` `\doextrafeet` is the code extending `\@makecol` to cater for the extra `reledmac` feet. We have two categories of extra footnotes. By default, we order the footnote inserts so that the regular footnotes of  $\TeX$  are first, then familiar familiar footnotes and finally the critical footnotes.

```

4394 \newcommand*{\l@ddextrafeet}{%
4395   \IfStrEq{familiar-critical}{\@fnpos}
4396     {\do@feetX\Xdo@feet}%
4397     {%
4398       \IfStrEq{critical-familiar}{\@fnpos}%
4399       {\Xdo@feet\do@feetX}%
4400       {\do@feetX\Xdo@feet}%
4401     }%
4402 }%
4403
4404 %

```

`\Xdo@feet` `\Xdo@feet` is the code extending `\@makecol` to cater for the extra critical feet.

```

4405 \newcommand*{\Xdo@feet}{%
4406   \setbox\@outputbox \vbox{%
4407     \unvbox\@outputbox
4408     \@opXfeet}}
4409 %

```

`\@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`.



```

4410 \newcommand\print@Xnotes[1]{%
4411   \csuse{#1footstart}{#1}%
4412   \csuse{#1footgroup}{#1}%
4413 }%
4414 %

```

We print all series of notes by looping on them. We check before printing them that they are not voided.

```

4415 \newcommand*\@opXfeet{%
4416   \unless\ifnocritical@%
4417   \gdef\firstXseries@{%
4418     \def\do##1{%
4419       \ifvoid\csuse{##1footins}\else%
4420         \global\skip\csuse{##1footins}=\csuse{Xbeforenotes@##1}%
4421         \global\advance\skip\csuse{##1footins} by\csuse{Xafterrule@##1}%
4422         \print@Xnotes{##1}%
4423       \fi%
4424     }%
4425     \dolistloop{\@series}%
4426   \fi%
4427 }%
4428 %

```

**\l@ddodoreinextrafeet** \l@ddodoreinextrafeet is the code for catering for the extra footnotes within \@reinserts. We use the same category and ordering as in \l@ddoxtrafeet.

```

4429 \newcommand*\l@ddodoreinextrafeet{%
4430   \IfStrEq{familiar-critical}{\@fnpos}
4431   {\@doreinfeetX\X@doreinfeet}%
4432   {%
4433     \IfStrEq{critical-familiar}{\@fnpos}%
4434     {\X@doreinfeet\@doreinfeetX}%
4435     {\@doreinfeetX\X@doreinfeet}%
4436   }%
4437 }
4438
4439 %

```

**\X@doreinfeet** \X@doreinfeet is the code for catering for the extra critical footnotes within \@reinserts.

```

4440 \newcommand*\X@doreinfeet{%
4441   \unless\ifnocritical@%
4442   \def\do##1{%
4443     \ifvoid\csuse{##1footins}\else%
4444       \insert\csuse{##1footins}{\unvbox\csuse{##1footins}}%
4445     \fi%
4446   \dolistloop{\@series}
4447   \fi%
4448 }
4449

```

```

4450 %

\print@notesX We have to add all the new kinds of familiar footnotes to the output routine. The normal
\do@feetX way to add one series. \print@Xnotes is replaced by reledpar when using \Pages.
\@doreinfeetX
4451 \newcommand\print@notesX[1]{%
4452 \csuse{footstart#1}{#1}%
4453 \csuse{footgroup#1}{#1}%
4454 }%
4455 %

We print all the series of notes by looping on them. We check before printing them that
they are not voided.

4456 \newcommand*\do@feetX{%
4457 \unless\ifnofamiliar%
4458 \gdef\firstseriesX@{}%
4459 \setbox\@outputbox \vbox{%
4460 \unvbox\@outputbox%
4461 \def\do##1{%
4462 \ifvoid\csuse{footins##1}\else%
4463 \global\skip\csuse{footins##1}=\csuse{beforenotesX@##1}%
4464 \global\advance\skip\csuse{footins##1} by\csuse{afterterruleX@##1}%
4465 \print@notesX{##1}%
4466 \fi%
4467 }%
4468 \dolistloop{\@series}}%
4469 \fi%
4470 }%
4471
4472 \newcommand{\@doreinfeetX}{%
4473 \unless\ifnofamiliar%
4474 \def\do##1{%
4475 \ifvoid\csuse{footins##1}\else
4476 \insert%
4477 \csuse{footins##1}
4478 {\unvbox\csuse{footins##1}}%
4479 \fi%
4480 }%
4481 \dolistloop{\@series}%
4482 \fi%
4483 }%
4484
4485 %

```

### 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.)

```

4486 \@ifclassloaded{memoir}{%
4487 %

memoir is loaded so we use memoir's built in hooks.

4488 \g@addto@macro{\m@mddoxtrafeet}{\l@ddoxtrafeet}%
4489 \g@addto@macro{\m@mddodoreinextrafeet}{\l@ddodoreinextrafeet}%
4490 }{%
4491 %

memoir has not been loaded, so patch \@makecol and \@reinserts.

4492 \@ifpackageloaded{fancyhdr}{%
4493   \patchcmd%
4494     {\latex@makecol}%
4495     {\xdef\@freelist{\@freelist\@midlist}}%
4496     {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4497     {}%
4498     {\led@error@fail@patch@makecol}%
4499   }{%
4500     \patchcmd%
4501       {\@makecol}%
4502       {\xdef\@freelist{\@freelist\@midlist}}%
4503       {\xdef\@freelist{\@freelist\@midlist}\l@ddoxtrafeet}%
4504       {}%
4505       {\led@error@fail@patch@makecol}%
4506     }%
4507
4508   \patchcmd%
4509     {\@reinserts}%
4510     {\ifvbox}%
4511     {\l@ddodoreinextrafeet\ifvbox}%
4512     {}%
4513     {\led@error@fail@patch@reinserts}%
4514   }
4515
4516 %

```

It turns out that `\@doclearpage` also needs modifying.

`\if@led@nofoot` We have to check if there are any leftover feet.

```

4517 \newif\if@led@nofoot
4518
4519 %

4520 \@ifclassloaded{memoir}{%
4521 %

```

If the memoir class is loaded we hook into its modified `\@doclearpage`.

```

\@mem@extranofeet22 \g@addto@macro{\@mem@extranofeet}{%%
4523 \def\do#1{%
4524 \unless\ifnocritical@%
4525 \ifvoid\csuse{#1footins}\else\@mem@nofootfalse\fi%
4526 \fi%
4527 \unless\ifnofamiliar@%
4528 \ifvoid\csuse{footins#1}\else\@mem@nofootfalse\fi%
4529 \fi%
4530 }
4531 \dolistloop{\@series}%
4532 }%
4533 }{%
4534 %

```

As memoir is not loaded we have patch \@doclearpage.

```

\@led@testifnofoot35 \newcommand*{\@led@testifnofoot}{%
\@doclearpage36 \@led@nofoottrue%
4537 \ifvoid\footins\else%
4538 \@led@nofootfalse%
4539 \fi%
4540 \def\do##1{%
4541 \unless\ifnocritical@%
4542 \ifvoid\csuse{##1footins}\else%
4543 \@led@nofootfalse%
4544 \fi%
4545 \fi%
4546 \unless\ifnofamiliar@%
4547 \ifvoid\csuse{footins##1}\else%
4548 \@led@nofootfalse%
4549 \fi%
4550 \fi%
4551 }%
4552 \dolistloop{\@series}%
4553 }%
4554
4555 \pretocmd%
4556 {\@doclearpage}%
4557 {\@led@testifnofoot}%
4558 {}%
4559 {\led@error@fail@patch@@doclearpage}%
4560
4561 \patchcmd%
4562 {\@doclearpage}%
4563 {\ifvoid\footins}%
4564 {\if\@led@nofoot}%
4565 {}%
4566 {\led@error@fail@patch@@doclearpage}%
4567

```

```

4568 }
4569
4570 %

```

## 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 saying `\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.

```

4571 \list@create{\labelref@list}
4572 %

```

`\zz@@@` A convenience macro to zero two labeling counters in one go.

```

4573 \newcommand*{\zz@@@}{000|000} % set two counters to zero in one go
4574
4575 %

```

`\edlabel` The `\edlabel` command first writes a `\@lab` macro to the `\linenum@out` file. It then checks to see that the `\labelref@list` actually has something in it (if not, it creates a dummy entry), and pops the next value for the current label, storing it in `\label@refs`. Finally it defines the label to be `\empty` so that any future check will turn up the fact that it has been used.<sup>31</sup>

This version of the original `edmac \label` uses `\@bsphack` and `\@esphack` to eliminate extra space problems and also use the  $\TeX$  write methods for the `.aux` file.

Jesse Billett<sup>32</sup> found that the original code could be off by several pages. This version, hopefully cures that, and also allows for non-arabic page numbering.

```

4576 \newcommand*{\edlabel}[1]{%
4577   \ifl@dpairing\ifautopar%
4578     \strut%
4579     \fi\fi%
4580     \@bsphack%
4581     \ifledRcol%

```

<sup>31</sup>The remaining macros in this section were kindly revised by Wayne Sullivan, who substantially improved their efficiency and flexibility.

<sup>32</sup>(jdb43@cam.ac.uk) via the ctt thread ‘ledmac cross referencing’, 25 August 2003.

```

4582 \write\linenum@outR{\string\@lab}%
4583 \ifx\labelref@listR\empty%
4584 \xdef\label@refs{\zz@@@}%
4585 \else%
4586 \glp\labelref@listR\to\label@refs%
4587 \fi%
4588 \ifvmode%
4589 \advance\label@refs%
4590 \fi%
4591 %

```

Use code from the kernel `\label` command to write the correct page number. Also define an `hypertarget` if `hyperref` package is loaded.

```

4592 \protected@write\@auxout{%
4593 {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR
4594 |{#1}}%
4595 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}}%
4596 \else%
4597 \write\linenum@out{\string\@lab}%
4598 \ifx\labelref@list\empty%
4599 \xdef\label@refs{\zz@@@}%
4600 \else%
4601 \glp\labelref@list\to\label@refs%
4602 \fi%
4603 \ifvmode%
4604 \advance\label@refs%
4605 \fi%
4606 \protected@write\@auxout{%
4607 {\string\l@dmake@labelsR\space\thepage|\label@refs|\the\c@pstartR|{#1}}%
4608 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1}}}{}}%
4609 \fi%
4610 \@esphack}%
4611 %

```

`\advance\label@refs` In cases where `\edlabel` is the first element in a paragraph, we have a problem with line counts, because line counts change only at the first horizontal box of the paragraph. Hence, we need to test `\edlabel` if it occurs at the start of a paragraph. To do so, we use `\ifvmode`. If the test is true, we must advance by one unit the amount of text we write into the `.aux` file. We do so using `\advance\label@refs` command.

```

4612 \newcounter{line}%
4613 \newcounter{subline}%
4614 \newcommand{\advance\label@refs}{%
4615 \setcounter{line}{\expandafter\labelrefsparseline\label@refs}%
4616 \stepcounter{line}%
4617 \ifsublines@%
4618 \setcounter{subline}{\expandafter\labelrefsparsesubline\label@refs}%
4619 %

```

```

4619     \stepcounter{subline}{1}%
4620     \def\label@refs{\theline\thesubline}%
4621     \else%
4622     \def\label@refs{\theline|0}%
4623     \fi%
4624 }
4625 \def\labelrefsparseline#1|#2{#1}
4626 \def\labelrefsparsesubline#1|#2{#2}
4627 %

```

**\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.

```

4628 \newcommand*{\l@make@labels}{%
4629 \def\l@make@labels#1|#2|#3|#4|#5{%
4630 \expandafter\ifx\csname the@label#5\endcsname \relax\else
4631 \led@warn@DuplicateLabel{#5}%
4632 \fi
4633 \expandafter\gdef\csname the@label#5\endcsname{#1|#2|#3|#4}%
4634 \ignorespaces}
4635 %
4636 %

```

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.

```

4637 \AtBeginDocument{%
4638 \def\l@make@labels#1|#2|#3|#4|#5{%
4639 }
4640 %
4641 %

```

**\@lab** The `\@lab` command, which appears in the `\linenum@out` file, appends the current values of page, line and sub-line to the `\labelref@list`. These values are defined by the earlier `\@page`, `\@nl`, and the `\sub@on` and `\sub@off` commands appearing in the `\linenum@out` file.

TeX uses the page counter for page numbers. However, it appears that this is not the right place to grab the page number. That task is now done in the `\edlabel` macro. This version of `\@lab` appends just the current line and sub-line numbers to `\labelref@list`.

```

4642 \newcommand*{\@lab}{%
4643 \ifledRcol
4644

```

```

4645 \xright@appenditem{\linenumr@p{\line@numR}}|{%
4646 \ifsublines@ \sublinenumr@p{\subline@numR}\else 0\fi}%
4647 \to\labelref@listR
4648 \else
4649 \xright@appenditem{\linenumr@p{\line@num}}|{%
4650 \ifsublines@ \sublinenumr@p{\subline@num}\else 0\fi}%
4651 \to\labelref@list
4652 \fi}
4653 %

```

`\applabel` `\applabel`, if called in `\edtext` will insert automatically both a start and an end label for the current edtext lines.

```

4654 \newcommand*{\applabel}[1]{%
4655 \ifnum\@edtext@level>0%
4656 %

```

Label should not be already defined.

```

4657 \ifcsundef{the@label#1}{%
4658 \csdef{the@label#1}{applabel}%
4659 }%
4660 {%
4661 \led@warn@DuplicateLabel{#1 (applabel)}%
4662 }%
4663 %

```

Parse the `\edtext` line numbers.

```

4664 \expandafter\l@dp@rsefootspec\l@d@nums|{%
4665 %

```

Use the  $\TeX$  standard hack for label.

```

4666 \@bsphack%
4667 %

```

And now, write the data in the auxiliary file.

```

4668 \ifledRcol%
4669 \protected@write\@auxout{%
4670 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
4671 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4672 \protected@write\@auxout{%
4673 {\string\l@dmake@labelsR\space\l@dparsedendpage|\l@dparsedendline|\l@dparsedendsub|\the\c@pstartR|{#1:end}}}%
4674 \else%
4675 \protected@write\@auxout{%
4676 {\string\l@dmake@labelsR\space\l@dparsedstartpage|\l@dparsedstartline|\l@dparsedstartsub|\the\c@pstartR|{#1:start}}%
4677 \ifdef{\hypertarget}{\Hy@raisedlink{\hypertarget{#1:start}}}{}}}%
4678 \protected@write\@auxout{%

```



```

4679         {\string\l@dmake@labels\space\l@dparsedendpage|\l@dparsedendline
|\l@dparsedendsub|\the\c@pstart|{#1:end}}}%
4680         \fi%
4681 %

```

Use the  $\TeX$  standard hack for label.

```

4682         \esphack%
4683 %

```

Warning if `\applabel` is called outside of `\edtext`.

```

4684     \else%
4685         \led@warn@AppLabelOutEdtext{#1}%
4686     \fi%
4687 %

```

End of `\applabel`

```

4688 }%
4689 %

```

**`\wrap@edcrossref`** `\wrap@edcrossref` is called around all `reledmac` crossref commands, except those which start with `x`. It adds the hyperlink.

```

4690 \newrobustcmd{\wrap@edcrossref}[2]{%
4691     \ifdef{\hyperlink}%
4692         {\hyperlink{#1}{#2}}%
4693     {#2}%
4694 }
4695 %

```

**`\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`.

```

4696 \newcommand*{\edpageref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{1}{#1}}}
4697 \newcommand*{\xpageref}[1]{\l@dgetref@num{1}{#1}}
4698 %
4699 %

```

**`\edlineref`** If the specified label exists, `\lineref` gives its line number.

**`\xlineref`**

```

4700 \newcommand*{\edlineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{2}{#1}}}%
4701 \newcommand*{\xlineref}[1]{\l@dgetref@num{2}{#1}}%
4702 %
4703 %

```

**\sublineref** If the specified label exists, \sublineref gives its sub-line number.

**\xsublineref**

```

4704 \newcommand*\sublineref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{3}{#1}}}
4705 \newcommand*\xsublineref}[1]{\l@dgetref@num{3}{#1}}
4706
4707 %

```

**\pstarteref** If the specified label exists, \pstarteref gives its pstart number.

**\xpstarteref**

```

4708 \newcommand*\pstarteref}[1]{\l@dref@undefined{#1}\wrap@edcrossref{#1}{\l@dgetref@num{4}{#1}}}
4709 \newcommand*\xpstarteref}[1]{\l@dgetref@num{4}{#1}}
4710
4711 %

```

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.

```

4712 \newcommand*\l@dref@undefined}[1]{%
4713   \expandafter\ifx\csname the@label#1\endcsname\relax
4714     \led@warn@RefUndefined{#1}%
4715   \fi}
4716
4717 %

```

**\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.

```

4718 \newcommand*\l@dgetref@num}[2]{%
4719   \expandafter
4720   \ifx\csname the@label#2\endcsname \relax
4721     000%
4722   \else
4723     \expandafter\expandafter\expandafter
4724     \l@dlabel@parse\csname the@label#2\endcsname| #1%
4725   \fi}
4726
4727 %

```

`\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`.)

```

4728 \newcommand*{\l@dlabel@parse}{%
4729 \def\l@dlabel@parse#1|#2|#3|#4|#5{%
4730   \ifcase #5%
4731   \or #1%
4732   \or #2%
4733   \or #3%
4734   \or #4%
4735   \fi}
4736 %

```

`\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 can not 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.

```

4737 \newcommand*{\xxref}[2]{%
4738 {%
4739   \expandafter\ifx\csname the@label#1\endcsname \relax%
4740   \expandafter\let\csname the@@label#1\endcsname\zz@@@%
4741   \else%
4742   \expandafter\def\csname the@@label#1\endcsname{\l@dgetref@num
{1}{#1}|\l@dgetref@num{2}{#1}|\l@dgetref@num{3}{#1}}%
4743   \fi%
4744   \expandafter\ifx\csname the@label#2\endcsname \relax%
4745   \expandafter\let\csname the@@label#2\endcsname\zz@@@%
4746   \else%
4747   \expandafter\def\csname the@@label#2\endcsname{\l@dgetref@num
{1}{#2}|\l@dgetref@num{2}{#2}|\l@dgetref@num{3}{#2}}%
4748   \fi%
4749   \ifdefined\@Rlineflag%
4750   \StrDel{\csuse{the@@label#1}}{\@Rlineflag}[\@tempa]%
4751   \StrDel{\csuse{the@@label#2}}{\@Rlineflag}[\@tempb]%
4752   \else%
4753   \letcs{\@tempa}{the@@label#1}%
4754   \letcs{\@tempb}{the@@label#2}%
4755   \fi%
4756   \linenum{\@tempa|%

```

```

4757 \@tempb}}}%
4758
4759 %

```

`\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` 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`.

```

4760 \xdef\Xtwolines@appref{}%
4761 \xdef\Xmorethantwolines@appref{}%
4762 \newtoggle{Xtwolinesbutnotmore@appref}%
4763 \newtoggle{Xtwolinesonlyinsamepage@appref}%
4764
4765 \xdef\Xendtwolines@apprefwithpage{}%
4766 \xdef\Xendmorethantwolines@apprefwithpage{}%
4767 \newtoggle{Xendtwolinesbutnotmore@apprefwithpage}%
4768 \newtoggle{Xendtwolinesonlyinsamepage@apprefwithpage}%
4769
4770 %

```

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.

```

4771
4772 \xdef\Xboxstartlinenum@appref{Opt}
4773 \xdef\Xboxendlinenum@appref{Opt}
4774
4775 \xdef\Xendboxstartlinenum@apprefwithpage{Opt}
4776 \xdef\Xendboxendlinenum@apprefwithpage{Opt}
4777
4778 %

```

Now, declare the default value of `\@apprefprefixsingle` and `\@apprefprefixmore`, and the commands which defines them

```

4779 \newcommand\@apprefprefixsingle{}%
4780 \newcommand\@apprefprefixmore{}%
4781
4782 \newcommand{\apprefprefixsingle}[1]{%
4783   \gdef\@apprefprefixsingle{#1}%
4784 }
4785
4786 \newcommand{\setapprefprefixmore}[1]{%

```

```

4787 \gdef\@apprefprefixmore{#1}%
4788 }
4789
4790 %

```

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.

```

4791 \newcommandx{\appref}[2][1,usedefault]{%
4792   \IfStrEq{#1}{fulllines}%
4793     {\toggletrue{fulllines@}}%
4794     {}%
4795   \xdef\@currentseries{appref}%
4796   \ifdefempty{\@apprefprefixmore}%
4797     {\@apprefprefixsingle}%
4798     {%
4799       \IfEq{\xlineref{#2:start}}{\xlineref{#2:end}}%
4800       {\@apprefprefixsingle}%
4801       {\@apprefprefixmore}%
4802     }%
4803   \printlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:start}
4804   |\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4805   \togglefalse{fulllines@}%
4806 }%
4807 % \changes{v1.23.0}{2015/05/18}{Debug \protect\cs{Xendtwolines}, \protect\
4808 %   cs{Xendmorethantwolines}, \protect\cs{Xendtwolinesbutnotmore} and \protect\
4809 %   cs{Xendtwolinesonlyinsamepage} when using \protect\cs{apprefwithpage}.}
4810 \newcommandx{\apprefwithpage}[2][1,usedefault]{%
4811   \IfStrEq{#1}{fulllines}%
4812     {\toggletrue{fulllines@}}%
4813     {}%
4814   \xdef\@currentseries{apprefwithpage}%
4815   \printendlines\xpageref{#2:start}|\xlineref{#2:start}|\xsublineref{#2:
4816   start}|\xpageref{#2:end}|\xlineref{#2:end}|\xsublineref{#2:end}||%
4817   \togglefalse{fulllines@}%
4818 }%
4819 %

```

`\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 say `\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`.

```

4817 \newcommand*{\edmakelabel}[2]{\expandafter\xdef\csname the@label#1\
4818   endcsname{#2}}

```

```
4818
4819 %
```

(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. 110 and V.9 p. 81), 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.

```
4820 \pretocmd{\@xympar}%
4821   {\ifnumberedpar@
4822     \led@warn@NoMarginpars
4823     \@esphack
4824     \else}%
4825   {}%
4826   {}%
4827
4828 \apptocmd{\@xympar}%
4829   {\fi}%
4830   {}
4831   {}
4832
4833 %
```

We provide side notes as replacement for `\marginpar` in numbered text.

`\sidenote@margin` These are the sidenote equivalents to `\line@margin` and `\linenummargin` for specifying which margin. The default is the right margin (opposite to the default for line numbers). `\l@dgetsidenote@margin` returns the number associated to side note margin:

**left** : 0

**right** : 1

**outer** : 2

**inner** : 3

```
4834 \newcount\sidenote@margin
4835 \newcommand*\sidenotemargin[1]{\{%
4836   \l@dgetsidenote@margin{#1}%
4837   \ifnum\l@dttempcntb>\m@ne
```

```

4838 \ifledRcol
4839 \global\sidenote@marginR=\@l@tempcntb
4840 \else
4841 \global\sidenote@margin=\@l@tempcntb
4842 \fi
4843 \fi}}
4844 \newcommand*{\l@dgetsidenote@margin}[1]{%
4845 \def\@tempa{#1}\def\@tempb{left}%
4846 \ifx\@tempa\@tempb
4847 \@l@tempcntb \z@
4848 \else
4849 \def\@tempb{right}%
4850 \ifx\@tempa\@tempb
4851 \@l@tempcntb \@ne
4852 \else
4853 \def\@tempb{outer}%
4854 \ifx\@tempa\@tempb
4855 \@l@tempcntb \tw@
4856 \else
4857 \def\@tempb{inner}%
4858 \ifx\@tempa\@tempb
4859 \@l@tempcntb \thr@@
4860 \else
4861 \led@warn@BadSidenotemargin
4862 \@l@tempcntb \m@ne
4863 \fi
4864 \fi
4865 \fi
4866 \fi}
4867 \sidenotemargin{right}
4868
4869 %

```

`\l@dlp@rbox` We need two boxes to store sidenote texts.

```

\l@drp@rbox
4870 \newbox\l@dlp@rbox
4871 \newbox\l@drp@rbox
4872
4873 %

```

`\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
4874 \newdimen\ledlsnotewidth \ledlsnotewidth=\marginparwidth
\ledrsnotesep
4875 \newdimen\ledrsnotewidth \ledrsnotewidth=\marginparwidth
\ledlsnotefontsetup
4876 \newdimen\ledlsnotesep \ledlsnotesep=\linenumsep
\ledrsnotefontsetup
4877 \newdimen\ledrsnotesep \ledrsnotesep=\linenumsep
4878 \newcommand*{\ledlsnotefontsetup}{\raggedleft\footnotesize}
4879 \newcommand*{\ledrsnotefontsetup}{\raggedright\footnotesize}
4880

```

```

4881 %

\ledleftnote \ledleftnote, \ledrightnote, \ledinnernote, \ledouternote are the user com-
\ledrightnote mands for left, right, inner and outer sidenotes. The two last one are just alias for the
\ledinnernote two first one, depending of the page number. \ledsidenote{<text>} is the command
\ledouternote for a moveable sidenote.
\ledsidenote
4882 \newcommand*{\ledleftnote}[1]{\edtext{}{\l@dlsnote{#1}}}
4883 \newcommand*{\ledrightnote}[1]{\edtext{}{\l@drsnote{#1}}}
4884
4885 \newcommand*{\ledinnernote}[1]{%
4886 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
4887 \ledleftnote{#1}%
4888 \else%
4889 \ledrightnote{#1}%
4890 \fi%
4891 }
4892
4893 \newcommand*{\ledouternote}[1]{%
4894 \ifodd\c@page% Do not use \page@num, because it is not yet calculated
when command is called
4895 \ledrightnote{#1}%
4896 \else%
4897 \ledleftnote{#1}%
4898 \fi%
4899 }
4900
4901 \newcommand*{\ledsidenote}[1]{\edtext{}{\l@dcsnote{#1}}}
4902 %

```

**\l@dlsnote** . The ‘footnotes’ for left, right, and moveable sidenotes. The whole scheme is reminiscent of the critical footnotes code.

```

\l@drsnote
\l@dcsnote
4903 \newif\ifrighnoteup
4904 \righnoteuptrue
4905
4906 \newcommand*{\l@dlsnote}[1]{%
4907 \begingroup%
4908 \newcommand{\content}{#1}%
4909 \ifnumberedpar@
4910 \ifledRcol%
4911 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
4912 \to\inserts@listR
4913 \global\advance\insert@countR \@ne%
4914 \else%
4915 \xright@appenditem{\noexpand\l@dlsnote{\expandonce\content}}{%
4916 \to\inserts@list
4917 \global\advance\insert@count \@ne%
4918 \fi

```



```

4919 \fi\ignorespaces\endgroup}
4920
4921 \newcommand*{\l@drsnote}[1]{%
4922 \begingroup%
4923 \newcommand{\content}{#1}%
4924 \ifnumberedpar@
4925 \ifledRcol%
4926 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
4927 \to\inserts@listR
4928 \global\advance\insert@countR \@ne%
4929 \else%
4930 \xright@appenditem{\noexpand\l@drsnote{\expandonce\content}}{%
4931 \to\inserts@list
4932 \global\advance\insert@count \@ne%
4933 \fi
4934 \fi\ignorespaces\endgroup}
4935
4936 \newcommand*{\l@dcnote}[1]{%
4937 \begingroup%
4938 \newcommand{\content}{#1}%
4939 \ifnumberedpar@
4940 \ifledRcol%
4941 \xright@appenditem{\noexpand\l@dcnote{\expandonce\content}}{%
4942 \to\inserts@listR
4943 \global\advance\insert@countR \@ne%
4944 \else%
4945 \xright@appenditem{\noexpand\l@dcnote{\expandonce\content}}{%
4946 \to\inserts@list
4947 \global\advance\insert@count \@ne%
4948 \fi
4949 \fi\ignorespaces\endgroup}
4950
4951 %

```

**\vl@dlsnote** Put the left/right text into boxes, but just save the moveable text. **\l@dcnotetext**, **\vl@drsnote** **\l@dcnotetext@l** and **\l@dcnotetext@r** are etoolbox's lists which will store the content of side notes. We store the content in lists, because we need to loop later on them, in case many sidenote co-exist for the same line. That is there some special test to do, in order to:

- Store the content of **\ledsidenote** to **\l@dcnotetext** in any cases.
- Store the content of **\rightsidenote** to:
  - **\l@dcnotetext** if **\ledsidenote** is to be put on right.
  - **\l@dcnotetext@r** if **\ledsidenote** is to be put on left.
- Store the content of **\leftsidenote** to:
  - **\l@dcnotetext** if **\ledsidenote** is to be put on left.

– \l@dcstotetext@l if \ledsidenote is to be put on right.

```

4952 \newcommand*{\vl@dlsnote}[1]{%
4953   \ifledRcol@%
4954     \@l@dttempcntb=\sidenote@marginR%
4955     \ifnum\@l@dttempcntb>\@ne%
4956       \advance\@l@dttempcntb by\page@numR%
4957     \fi%
4958   \else%
4959     \@l@dttempcntb=\sidenote@margin%
4960     \ifnum\@l@dttempcntb>\@ne%
4961       \advance\@l@dttempcntb by\page@num%
4962     \fi%
4963   \fi%
4964   \ifodd\@l@dttempcntb%
4965     \listgadd{\l@dcstotetext@l}{#1}%
4966   \else%
4967     \listgadd{\l@dcstotetext}{#1}%
4968   \fi
4969 }
4970 \newcommand*{\vl@drsnote}[1]{%
4971   \ifledRcol@%
4972     \@l@dttempcntb=\sidenote@marginR%
4973     \ifnum\@l@dttempcntb>\@ne%
4974       \advance\@l@dttempcntb by\page@numR%
4975     \fi%
4976   \else%
4977     \@l@dttempcntb=\sidenote@margin%
4978     \ifnum\@l@dttempcntb>\@ne%
4979       \advance\@l@dttempcntb by\page@num%
4980     \fi%
4981   \fi%
4982   \ifodd\@l@dttempcntb%
4983     \listgadd{\l@dcstotetext}{#1}%
4984   \else%
4985     \listgadd{\l@dcstotetext@r}{#1}%
4986   \fi%
4987 }
4988 \newcommand*{\vl@dcsnote}[1]{\listgadd{\l@dcstotetext}{#1}}
4989
4990 %

```

`\setl@dlp@rbox` \setl@dlprbox{<lednums>}{<tag>}{<text>} puts <text> into the \l@dlp@rbox box.  
`\setl@drpr@box` And similarly for the right side box. It is these boxes that finally get displayed in the margins.

```

4991 \newcommand*{\setl@dlp@rbox}[1]{%
4992   {\parindent\z@\hspace=\ledlsnotewidth\ledlsnotefontsetup
4993     \global\setbox\l@dlp@rbox
4994     \ifleftnoteup

```

```

4995     =\vbox to\z@{\vss #1}%
4996     \else
4997     =\vbox to 0.70\baselineskip{\strut#1\vss}%
4998     \fi}}
4999 \newcommand*\setl@drp@rbox}[1]{%
5000 {\parindent\z@ \hsize=\ledrsnotewidth\ledrsnotefontsetup
5001  \global\setbox\l@drp@rbox
5002  \ifrightnoteup
5003   =\vbox to\z@{\vss#1}%
5004  \else
5005   =\vbox to0.7\baselineskip{\strut#1\vss}%
5006  \fi}}
5007 \newif\ifleftnoteup
5008 \leftnoteuptrue
5009 %

```

**\@sidenotesep** This macro is used to separate sidenotes of the same line.

```

5010 \newcommand{\setsidenotesep}[1]{\gdef\@sidenotesep{#1}}
5011 \newcommand{\@sidenotesep}{, }
5012 %

```

**\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.

```

5013 \newcommand*\affixside@note){%
5014   \def\sidenotecontent@{}%
5015   \numgdef\itemcount@{0}%
5016   \def\do##1{%
5017     \ifnumequal{\itemcount@}{0}%
5018     {%
5019       \appto\sidenotecontent@{##1}}% Not print not separator before
the 1st note
5020     {\appto\sidenotecontent@{\@sidenotesep ##1}%
5021     }%
5022     \numgdef\itemcount@{\itemcount@+1}%
5023   }%
5024   \dolistloop{\l@dcstotetext}%
5025   \ifnumgreater{\itemcount@}{1}{\led@err@ManySidenotes}{}%
5026 %

```

And we do the same for left and right notes (not movable).

```

5027 \gdef\@templ@d{%
5028 \gdef\@templ@n{\l@dcstotetext\l@dcstotetext@1\l@dcstotetext@r}%
5029 \ifx\@templ@d\@templ@n \else%
5030 \if@twocolumn%

```

```

5031 \if@firstcolumn%
5032 \setl@dlp@rbox{##1}{\sidenotecontent@}%
5033 \else%
5034 \setl@drp@rbox{\sidenotecontent@}%
5035 \fi%
5036 \else%
5037 \l@dttempcntb=\sidenote@margin%
5038 \ifnum\l@dttempcntb>\@ne%
5039 \advance\l@dttempcntb by\page@num%
5040 \fi%
5041 \ifodd\l@dttempcntb%
5042 \setl@drp@rbox{\sidenotecontent@}%
5043 \gdef\sidenotecontent@{}%
5044 \numgdef{\itemcount@}{0}%
5045 \dolistloop{\l@dcstotext@1}%
5046 \ifnumgreater{\itemcount@}{1}{\led@err@ManyLeftnotes}{}%
5047 \setl@dlp@rbox{\sidenotecontent@}%
5048 \else%
5049 \setl@dlp@rbox{\sidenotecontent@}%
5050 \gdef\sidenotecontent@{}%
5051 \numgdef{\itemcount@}{0}%
5052 \dolistloop{\l@dcstotext@r}%
5053 \ifnumgreater{\itemcount@}{1}{\led@err@ManyRightnotes}{}%
5054 \setl@drp@rbox{\sidenotecontent@}%
5055 \fi%
5056 \fi%
5057 \fi%
5058 }
5059 %

```

## 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 `\@iiminipage` and `\endminipage` macros. We will arrange this so that additional series can be easily added.

`\l@dfteetbeginmini` These will be the hooks in `\@iiminipage` and `\endminipage`.  
`\l@dfteetendmini` They can be extended to handle other things if necessary.

```

5060 \ifnoledgroup@ \else%
5061 \newcommand*{\l@dfteetbeginmini}{\l@dedbeginmini\l@dfambeginmini}
5062 \newcommand*{\l@dfteetendmini}{%
5063 \IfStrEq{critical-familiar}{\@mpfnpos}%
5064 {\l@dedendmini\l@dfamendmini}%
5065 {%
5066 \IfStrEq{familiar-critical}{\@mpfnpos}%
5067 {\l@dfamendmini\l@dedendmini}%
5068 {\l@dedendmini\l@dfamendmini}%

```

```

5069     }%
5070   }%
5071 %

```

**\l@dedbeginmini** These handle the initiation and closure of critical footnotes in a minipage environment.

**\l@dedendmini**

```

5072 \newcommand*{\l@dedbeginmini}{%
5073   \unless\ifnocritical@%
5074     \def\do##1{\csletcs{v##1footnote}{mpv##1footnote}}%
5075     \dolistloop{\@series}%
5076   \fi%
5077 }
5078 \newcommand*{\l@dedendmini}{%
5079   \unless\ifnocritical@%
5080     \ifl@dpairing%
5081       \ifledRcol%
5082         \flush@notesR%
5083       \else%
5084         \flush@notes%
5085       \fi%
5086     \fi
5087     \def\do##1{%
5088       \ifvoid\csuse{mp##1footins}\else%
5089         \ifl@dpairing\ifparledgroup%
5090           \ifledRcol%
5091             \dingdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
5092 skip\@nameuse{mp##1footins}}%
5093           \else%
5094             \dingdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL
5095 +\skip\@nameuse{mp##1footins}}%
5096           \fi%
5097         \fi\fi%
5098         \csuse{mp##1footgroup}{##1}%
5099       \fi}%
5100     \dolistloop{\@series}%
5101   \fi%
5102 }%
5103 %

```

**\l@dfambeginmini** These handle the initiation and closure of familiar footnotes in a minipage environment.

**\l@dfamendmini**

```

5103 \newcommand*{\l@dfambeginmini}{%
5104   \unless\ifnofamiliar@%
5105     \def\do##1{\csletcs{vfootnote##1}{mpvfootnote##1}}%
5106     \dolistloop{\@series}%
5107   \fi%
5108 }%
5109
5110 \newcommand*{\l@dfamendmini}{%

```

```

5111 \unless\ifnofamiliar@%
5112   \def\do##1{%
5113     \ifvoid\csuse{mpfootins##1}\else%
5114     \csuse{mpfootgroup##1}{##1}%
5115   \fi}%
5116   \dolistloop{\@series}%
5117 \fi%
5118 }%
5119 %

```

`\@iiiminipage` This is our extended form of the kernel `\@iiiminipage` defined in `ltboxes.dtx`.

```

5120 \patchcmd%
5121   {\@iiiminipage}%
5122   {\let\@footnotetext\@mpfootnotetext}%
5123   {\let\@footnotetext\@mpfootnotetext\l@dfeetbeginmini}%
5124   {}%
5125   {\led@error@fail@patch@\@iiiminipage}%
5126 %

```

`\endminipage` This is our extended form of the kernel `\endminipage` defined in `ltboxes.dtx`.

```

5127 \patchcmd%
5128   {\endminipage}%
5129   {\footnoterule}%
5130   {\footnoterule\l@advance@parledgroup@beforenormalnotes}%
5131   {}%
5132   {\led@error@fail@patch@\endminipage}%
5133
5134 \patchcmd%
5135   {\endminipage}%
5136   {\@minipagefalse}%
5137   {\l@dfeetendmini\@minipagefalse}%
5138   {}%
5139   {\led@error@fail@patch@\endminipage}%
5140
5141 %

```

`\l@dunboxmpfoot` `\l@dunboxmpfoot` insert normal footnotes for `ledgroup`.  
`\l@advance@parledgroup@beforenormalnotes`

```

5142 \newcommand*{\l@dunboxmpfoot}{%
5143   \vskip\skip\@mpfootins
5144   \normalcolor
5145   \footnoterule
5146   \l@advance@parledgroup@beforenormalnotes
5147   \unvbox\@mpfootins%
5148 }
5149 %

```

When using parallel `ledgroup`, we need to store the vertical space added before footnote, in order to compensate them between left and right pages.

```

5150 \newcommand{\l@advance@parledgroup@beforenormalnotes}{%
5151   \ifparledgroup
5152     \ifl@dpairing
5153       \ifledRcol
5154         \dimdef{\parledgroup@beforenotesR}{\parledgroup@beforenotesR+
skip\@mpfootins}
5155       \else
5156         \dimdef{\parledgroup@beforenotesL}{\parledgroup@beforenotesL+
skip\@mpfootins}
5157       \fi
5158     \fi
5159   \fi
5160 }
5161 %

```

`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.

```

5162 \newenvironment{ledgroup}{%
5163   \resetprevpage@num%
5164   \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@%
5165   \let\@footnotetext\@mpfootnotetext
5166   \l@dfetbeginmini%
5167 }{%
5168   \par
5169   \unskip
5170   \ifvoid\@mpfootins\else
5171     \l@dunboxmpfoot
5172   \fi
5173   \l@dfetendmini%
5174 }
5175 %
5176 %

```

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

This environment puts footnotes at the end, even if that happens to be in the middle of a page, or crossing a page boundary. It is a sort of unboxed, variable `\langle width \rangle` minipage. The optional `\langle pos \rangle` controls the sideways position of numbered text.

```

5177 \newenvironment{ledgroupsize}[2][1]{%
5178 %

```

Set the various text measures.

```

5179 \hspace #2\relax
5180 %% \textwidth #2\relax
5181 %% \columnwidth #2\relax
5182 %

```

Initialize fills for centering.

```

5183 \let\ledllfill\hfil
5184 \let\ledrlfill\hfil
5185 \def\@tempa{#1}\def\@tempb{1}%
5186 %

```

Left adjusted numbered lines

```

5187 \ifx\@tempa\@tempb
5188 \let\ledllfill\relax
5189 \else
5190 \def\@tempb{r}%
5191 \ifx\@tempa\@tempb
5192 %

```

Right adjusted numbered lines

```

5193 \let\ledrlfill\relax
5194 \fi
5195 \fi
5196 %

```

Set up the footnoting.

```

5197 \def\@mpfn{\mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
5198 \let\@footnotetext\@mpfootnotetext
5199 \l@dfetbeginmini%
5200 }{%
5201 \par
5202 \unskip
5203 \ifvoid\@mpfootins\else
5204 \l@dunboxmpfoot
5205 \fi
5206 \l@dfetendmini%
5207 }
5208
5209 %

```

Close the \ifnoledgroup@else.

```

5210 \fi%
5211 %

```

`\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`.

```

5212 \newif\ifledgroupnotesL@
5213 \newif\ifledgroupnotesR@
5214 %

```

## XXVI Indexing

Here is some code for indexing using page and line numbers.

First, ensure that `imakeidx` or `indextools` is loaded *before* `eledmac`.



```

5215 \AtBeginDocument{%
5216   \unless\ifl@imakeidx%
5217     \@ifpackageloaded{imakeidx}{\led@error@ImakeidxAfterEledmac}{}%
5218   \fi%
5219   \unless\ifl@indextools%
5220     \@ifpackageloaded{indextools}{\led@error@indextoolsAfterEledmac}{}%
5221   \fi%
5222 }
5223 %

```

`\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
5224 \newcommand{\pagelinesep}{-}
5225 \newcommand{\edindexlab}{$&}
5226 \newcounter{labidx}
5227 \setcounter{labidx}{0}
5228
5229 %

```

`\doedindexlabel` This macro sets an `\edlabel`.

```

5230 \newcommand{\doedindexlabel}{%
5231   \stepcounter{labidx}%
5232   \edlabel{\edindexlab\thelabidx}%
5233 }
5234
5235 %

```

`\thepageline` This macro makes up the page/line number combo from the label/ref.

```

5236 \newcommand{\thepageline}{%
5237   \thepage%
5238   \pagelinesep%
5239   \xlineref{\edindexlab\thelabidx}%
5240 }
5241 %

```

`\thestartpageline` These macros make up the page/line start/end number when the `\edindex` command  
`\theendpageline` is called in critical notes.

```

5242 \newcommand{\thestartpageline}{%
5243   \l@dparsedstartpage%
5244   \pagelinesep%
5245   \l@dparsedstartline%
5246 }
5247 \newcommand{\theendpageline}{%
5248   \l@dparsedendpage%
5249   \pagelinesep%
5250   \l@dparsedendline%

```

```
5251 }
5252 %
```

**\if@edindex@fornote@true** This boolean test is switching at the beginning of each critical note, to allow index referring to this note.

```
5253 \newif\if@edindex@fornote@
5254 %
```

**\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 |.

```
5255 \newcommand{\prepare@edindex@fornote}[1]{%
5256     \l@dp@rsefootspec#1|}%
5257     \@edindex@fornote@true%
5258 }
5259 %
```

**\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.

```
5260 \newcommand{\get@edindex@ledinnote@command}{%
5261     \ifxindy%
5262         \gdef\@ledinnote@command{%
5263             ledinnote\thelabidx%
5264         }%
5265         \ifxindyhyperref%
5266             \immediate\write\eledmac@xindy@out{%
5267                 (define-attributes ("ledinnote\thelabidx"))^^J
5268                 \space\space(markup-locref^^J
5269                 \eledmacmarkuplocorefdepth^^J
5270                 :open "\string\ledinnote[\edindexlab\thelabidx]{\@index@command
5271             }{"^^J
5272                 :close "}"^^J
5273                 :attr "ledinnote\thelabidx"^^J
5274             )
5275         }%
5276     \else%
5277         \immediate\write\eledmac@xindy@out{%
5278             (define-attributes ("ledinnote\thelabidx"))^^J
5279             \space\space(markup-locref^^J
5280             \eledmacmarkuplocorefdepth^^J
5281             :open "\string\ledinnote{\@index@command}{"^^J
5282             :close "}"^^J
5283             :attr "ledinnote\thelabidx"^^J
```

```

5283     )
5284   }%
5285   \fi%
5286 %

```

If we do not use xindy option, `\@ledinnote@command` will produce something like `ledinnote{formattingcommand}`.

```

5287 \else%
5288   \gdef\@ledinnote@command{%
5289     ledinnote[\edindexlab\thelabidx]{\@index@command}%
5290   }%
5291   \fi%
5292 }
5293 %

```

`\get@index@command` This macro is used to analyse if a text to be indexed has a command after a |.

```

5294 \def\get@index@command#1|#2+{%
5295   \gdef\@index@txt{#1}%
5296   \gdef\@index@command{#2}%
5297   \xdef\@index@parenthesis{}%
5298   \IfBeginWith{\@index@command}{(}{%
5299     \StrGobbleLeft{\@index@command}{1}{\@index@command@}%
5300     \global\let\@index@command\@index@command@%
5301     \xdef\@index@parenthesis{(}%
5302   }{}%
5303   \IfBeginWith{\@index@command}{)}{%
5304     \StrGobbleLeft{\@index@command}{1}{\@index@command@}%
5305     \global\let\@index@command\@index@command@%
5306     \xdef\@index@parenthesis{)}%
5307   }{}%
5308 }
5309 %

```

`\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`

```

5310 \newcommandx{\ledinnote}[3][1,usedefault]{%
5311   \ifboolexpr{%
5312     test{\ifdefequal{\iftrue}{\ifHy@hyperindex}}%
5313     or%
5314     bool {xindyhyperref@}%
5315   }%
5316   {%
5317     \csuse{#2}{\hyperlink{#1}{\ledinnotemark{#3}}}%
5318   }%
5319   {%
5320     \csuse{#2}{\ledinnotemark{#3}}%

```

```

5321 }%
5322 }%
5323 \newcommand{\ledinnothyperpage}[2]{\csuse{#1}{\ledinnotemark{\hyperpage
5324   {#2}}}}%
5325 \newcommand{\ledinnotemark}[1]{#1\emph{n}}%
5326 %

```

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
5326 \newcommandx{\@wredindex}[2][1=\expandonce\jobname,usedefault]{%#1 = the
5327   index name, #2 = the text
5328   \global\let\old@Rlineflag\@Rlineflag%
5329   \gdef\@Rlineflag{}%
5330   \ifl@imakeidx%
5331     \if@edindex@fornote%
5332       \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5333       \get@edindex@ledinnote@command%
5334       \expandafter\imki@wrindexentry{#1}{\@index@txt|(\@ledinnote@command
5335         }\the startpageline}%
5336       \expandafter\imki@wrindexentry{#1}{\@index@txt|)\@ledinnote@command
5337         }\the endpageline}%
5338     \else%
5339       \get@edindex@hyperref{#2}%
5340       \imki@wrindexentry{#1}{\@index@txt\@edindex@hyperref}{\the pageline}%
5341     \fi%
5342   \else%
5343     \if@edindex@fornote%
5344       \IfSubStr[1]{#2}{|}{\get@index@command#2+}{\get@index@command#2|+}%
5345       \get@edindex@ledinnote@command%
5346       \expandafter\protected@write\@indexfile{}%
5347       {\string\indexentry{\@index@txt|(\@ledinnote@command}{\the startpageline}
5348       }%
5349       \expandafter\protected@write\@indexfile{}%
5350       {\string\indexentry{\@index@txt|)\@ledinnote@command}{\the endpageline}
5351       }%
5352     \else%
5353       \protected@write\@indexfile{}%
5354       {\string\indexentry{#2}{\the pageline}
5355       }%
5356     \fi%
5357   \fi%

```

```

5355 \endgroup
5356 \global\let\@Rlineflag\old@Rlineflag%
5357 \@esphack%
5358 }
5359 %

```

Need to add the definition of `\edindex` to `\makeindex`, and initialise `\edindex` to do nothing.

```

5360 \pretocmd{\makeindex}{%
5361   \def\edindex{\@bsphack
5362     \doedindexlabel
5363     \begingroup
5364     \@sanitize
5365     \@wredindex}}{}{}
5366 \newcommand{\edindex}[1]{\@bsphack\@esphack}
5367 %

```

`\hyperlinkformat` `\hyperlinkformat` command is to be used to have both a internal hyperlink and a format, when indexing.

```

5368 \newcommand{\hyperlinkformat}[3]{%
5369   \ifstrepty{#1}%
5370     {\hyperlink{#2}{#3}}%
5371     {\csuse{#1}{\hyperlink{#2}{#3}}}%
5372   }%
5373 %

```

`\hyperlinkR` `\hyperlinkR` command is to be used to create a internal hyperlink and `\ledRflag`, when indexing.

```

5374 \newcommand{\hyperlinkR}[2]{%
5375   \hyperlink{#1}{#2\@Rlineflag}%
5376 }%
5377 %
5378 %

```

`\hyperlinkformatR` `\hyperlinkformatR` command is to be used to create a internal hyperlink, a format and a `\@Rlineflag`, when indexing.

```

5379 \newcommand{\hyperlinkformatR}[3]{%
5380   \hyperlinkformat{#1}{#2}{#3\@Rlineflag}%
5381 }%
5382 %
5383 %

```

`\get@edindex@hyperref` `\get@edindex@hyperref` is to be used to define the `\@edindex@hyperref` macro, which, in index, links to the point where the index was called (with `hyperref`).

```

5384 \newcommand{\get@edindex@hyperref}[1]{%
5385 %

```

We have to disable temporary spaces to work through a xstring bug (or feature?)

```

5386 \edef\temp@{%
5387 \catcode`\ =9 %space need for catcode
5388 #1%
5389 \catcode`\ =10 % space need for catcode
5390 }%
5391 %

```

Now, we define \@edindex@hyperref if the hyperindex of hyperref is enabled.

```

5392 \ifdefequal{\iftrue}{\ifHy@hyperindex}{%
5393 \IfSubStr{\temp@}{|}%
5394 {\get@index@command#1+%
5395 \ifledRcol%
5396 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5397 hyperlinkformatR{\@index@command}%
5398 {\edindexlab\thelabidx}}%
5399 \else%
5400 \gdef\@edindex@hyperref{|\@index@parenthesis %space kept
5401 hyperlinkformat{\@index@command}%
5402 {\edindexlab\thelabidx}}%
5403 \fi%
5404 }%
5405 {\get@index@command#1|+%
5406 \ifledRcol%
5407 \gdef\@edindex@hyperref{|\hyperlinkR{\edindexlab\thelabidx}}%
5408 \else%
5409 \gdef\@edindex@hyperref{|\hyperlink{\edindexlab\thelabidx}}%
5410 \fi%
5411 }%
5412 }%
5413 %

```

```

5414 % If we use both xindy and hyperref, first get the \protect\cs{
index@command} command.
5415 % Then define \protect\cs{@edindex@hyperref} in the form \verb+eledmacXXX+
5416 % \begin{macrocode}
5417 {\ifxindyhyperref%
5418 \IfSubStr{\temp@}{|}%
5419 {\get@index@command#1+%
5420 {\get@index@command#1|+%
5421 \gdef\@edindex@hyperref{|\eledmac\thelabidx}%
5422 %

```

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.

```

5423 \IfStrEq{\@index@parenthesis}{(}%
5424 {%
5425 \csxdef{xindy@parenthesis@{\@index@txt}{\thelabidx}%
5426 \gdef\@edindex@hyperref{|(eledmac\thelabidx}%

```

```

5427     }%
5428     {}%
5429 %

```

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`.

```

5430     \IfStrEq{\@index@parenthesis}{})}%
5431     {%
5432     \xdef\@edindex@hyperref{|\@edmac\csuse{xindy@parenthesis@
\@index@txt}}}%
5433     \global\csundef{xindy@parenthesis@\@index@txt}%
5434     }%
5435 %

```

Write in the `.xdy` file the attributes of the location.

```

5436     {%
5437     \immediate\write\@edmac\xindy@out{%
5438     (define-attributes ("eledmac\thelabidx"))^^J
5439     \space\space(markup-locref^^J
5440     \eledmacmarkuplocdepth^^J
5441     :open "\string\hyperlink%
5442     \ifledRcol R\fi%
5443     {\edindexlab\thelabidx}%
5444     {\ifdefempty{\@index@command}%
5445     }%
5446     {\@backslashchar\@index@command}%
5447     {"^^J
5448     :close "}}^^J
5449     :attr "eledmac\thelabidx"^^J
5450     )
5451     }%
5452     }%
5453 %

```

And now, in any other case.

```

5454     \else%
5455     \gdef\@index@txt{#1}%
5456     \gdef\@edindex@hyperref{}%
5457     \fi%
5458     }%
5459 }
5460 %

```

## 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.

```
5461 \def\@hangingsymbol{}
5462 \newcommand*\sethangingsymbol}[1]{%
5463   \gdef\@hangingsymbol{#1}%
5464 }%
5465 \newif\ifinstanza
5466 %
```

`\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.

```
5467 \newif\ifinserthangingsymbol
5468 \newcommand\inserthangingsymbol{%
5469   \ifinserthangingsymbol%
5470     \ifinstanza%
5471       \@hangingsymbol%
5472     \fi%
5473   \fi%
5474 }
5475 %
```

### 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.

```
5476 \newcommand*\ampersand{\char`\&}
5477
5478 %
```



### 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.

```
5479 \chardef\body=\catcode`\@
5480 \catcode`\@=11
5481 \chardef\next=\catcode`\&
5482 \catcode`\&=\active
5483
5484 %
```

### 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.

```
5485 \newcount\stanza@count
5486 \newlength{\stanzaindentbase}
5487 \setlength{\stanzaindentbase}{20pt}
5488
5489 %
```

`\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.

```
5490 \def\strip@szacnt#1,#2|{\def\@tempb{#1}\def\@tempa{#2|}}
5491 \newcommand*\setstanzavalues}[2]{\def\@tempa{#2,,|}%
5492   \stanza@count\z@
5493   \def\next{\expandafter\strip@szacnt\@tempa
5494     \ifx\@tempb\empty\let\next\relax\else
5495       \expandafter\mathchardef\csname #1@\number\stanza@count
5496         \@endcsname\@tempb\relax
5497       \advance\stanza@count\@ne\fi\next}%
5498   \next}
5499
5500 %
```

`\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).

```
5501 \newcommand*\setstanzaindents}[1]{\setstanzavalues{sza}{#1}}
5502 \newcommand*\setstanzapenalties}[1]{\setstanzavalues{szp}{#1}}
```

```
5503 %
5504 %
```

**\managestanza@modulo** Since version 0.13, the `stanzaindent`srepetition 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 `stanzaindent`srepetition counter, the command restarts it.

```
5505 \newcounter{stanzaindent}srepetition}
5506 \newcount\stanza@modulo
5507
5508 \newcommand*{\managestanza@modulo}[0]{
5509   \advance\stanza@modulo\@ne
5510   \ifnum\stanza@modulo>\value{stanzaindent}srepetition}
5511     \stanza@modulo\@ne
5512   \fi
5513 }
5514 %
```

**\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.

```
5515 \newcommand{\stanzaindent}[1]{%
5516   \hspace{\dimexpr#1\stanzaindentbase-\parindent\relax}%
5517   \ignorespaces%
5518 }%
5519 \WithSuffix\newcommand\stanzaindent*[1]{%
5520   \stanzaindent{#1}%
5521   \global\advance\stanza@modulo-\@ne%
5522   \ifnum\stanza@modulo=0%
5523     \global\stanza@modulo=\value{stanzaindent}srepetition}%
5524   \fi%
5525   \ignorespaces%
5526 }%
5527 %
```

## XXVII.5 Numbering stanza

Here, macro for numbering stanza. First, the stanza counter.

```
\thestanza28 \newcounter{stanza}
5529 \renewcommand{\thestanza}{%
5530   \textbf{\arabic{stanza}}%
5531 }
5532 %
```

`\ifnumberstanza` Then, macro to activate automatically numbering of stanza.

```
5533 \newif\ifnumberstanza%
5534 %
```

`\@insertstanzanumber` Now, macro called at the first line of of verse of a stanza.

```
5535 \newcommand{\@insertstanzanumber}[0]{%
5536   \ifnumberstanza%
5537     \ifl@dpairing%
5538       \ifledRcol%
5539         \stanzanumwrapper{\thestanzaR}%
5540       \else%
5541         \stanzanumwrapper{\thestanzaL}%
5542       \fi%
5543     \else%
5544       \stanzanumwrapper{\thestanza}%
5545     \fi%
5546     \setline{1}%
5547   \fi%
5548 }%
5549 %
```

`\@advancestanzanumber` Also a command to advance the counter of stanza.

```
5550 \newcommand{\@advancestanzanumber}[0]{%
5551   \ifnumberstanza%
5552     \ifl@dpairing%
5553       \ifledRcol%
5554         \addtocounter{stanzaR}{1}%
5555       \else%
5556         \addtocounter{stanzaL}{1}%
5557       \fi%
5558     \else%
5559       \addtocounter{stanza}{1}%
5560     \fi%
5561   \fi%
5562 }%
5563 %
```

`\stanzanumwrapper` And finally, the wrapper for stanza number

```
5564 \newcommand{\stanzanumwrapper}[1]{%
5565   \flagstanza{#1}%
5566 }%
5567 %
```

## XXVII.6 Stanza number in note

Here, the command called when printing stanza number in notes.

```

5568 \newcommand{\printstanza}[0]{%
5569     \ifboolexpr{bool{l@dpairing} or bool{l@dprintingpages} or bool{
l@dprintingcolumns}}{%
5570         \ifledRcol{%
5571             \thestanzaR%
5572         \else%
5573             \thestanzaL%
5574         \fi%
5575     }{%
5576         \thestanza%
5577     }%
5578 }
5579 %

```

## 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.

```

5580 \newcommandx{\stanza@line}[1][1]{
5581     \ifnum\value{stanzaindentrepetition}=0
5582         \parindent=\csname sza@\number\stanza@count
5583             @\endcsname\stanzaindentbase
5584     \else
5585         \parindent=\csname sza@\number\stanza@modulo
5586             @\endcsname\stanzaindentbase
5587         \managestanza@modulo
5588     \fi
5589     \pstart[#1]\stanza@hang\ignorespaces}
5590 \xdef\stanza@hang{\noexpand\leavevmode\noexpand\startlock
5591     \hangindent\expandafter
5592     \noexpand\csname sza@0@\endcsname\stanzaindentbase
5593     \hangafter\@ne}
5594 \def\sza@penalty{\count@\csname szp@\number\stanza@count @\endcsname
5595     \ifnum\count@>\@M\advance\count@-\@M\penalty-\else
5596     \penalty\fi\count@}
5597 %

```

`\@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 `\&`.

```

5598 \xdef\@startstanza[#1]{%
5599   \noexpand\instanzatrue\expandafter
5600   \begingroup%
5601   \catcode`\noexpand\&\active%
5602   \global\stanza@count\@ne\stanza@modulo\@ne
5603   \noexpand\ifnum\expandafter\noexpand
5604   \csname sz@0@\endcsname=\z@\let\noexpand\stanza@hang\relax
5605   \let\noexpand\endlock\relax\noexpand\else\interlinepenalty
5606   \@M\rightskip\z@ plus 1fil\relax\noexpand\fi\noexpand\ifnum
5607   \expandafter\noexpand\csname szp@0@\endcsname=\z@
5608   \let\noexpand\sza@penalty\relax\noexpand\fi%
5609   \def\noexpand&{%
5610     \noexpand\newverse[] []}%
5611   \def\noexpand\&{\noexpand\@stopstanza}%
5612   \noexpand\@advancestanzanumber%
5613   \noexpand\stanza@line[#1]\noexpand\@insertstanzanumber}
5614
5615 \newcommandx{\stanza}[1][1,usedefault]{\@startstanza[#1]}
5616
5617 \newcommandx{\@stopstanza}[1][1,usedefault]{%
5618   \unskip%
5619   \endlock%
5620   \pend[#1]%
5621   \endgroup%
5622   \instanzafalse%
5623 }
5624
5625 \newcommandx*{\newverse}[2][1,2,usedefault]{%
5626   \unskip%
5627   \endlock\pend[#1]\sza@penalty\global%
5628   \advance\stanza@count\@ne\stanza@line[#2]%
5629 }
5630
5631 %

```

**\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`.

```

5632 \newcommand*{\flagstanza}[2][\stanzaindentbase]{%
5633   \hskip -#1\llap{#2}\hskip #1\ignorespaces}
5634
5635 %

```

## 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.

```
5636 \catcode`\&=\next
5637 \catcode`\@=\body
5638 \setstanzavalues{szp}{0}
5639
5640 %
```

## 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.

```
5641 \newtoks\@emptytoks
5642
5643 %
```

The rest is from amsmath.

**\l@denvbody** A token register to contain the body.

```
5644 \newtoks\l@denvbody
5645
5646 %
```

**\addtol@denvbody** \addtol@denvbody{arg} adds arg to the token register \l@denvbody.

```
5647 \newcommand{\addtol@denvbody}[1]{%
5648   \global\l@denvbody\expandafter{\the\l@denvbody#1}}
5649
5650 %
```

`\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`.

```

5651 \newcommand{\l@dcollect@body}[1]{%
5652   \l@denbody{\expandafter#1\expandafter{\the\l@denbody}}%
5653   \edef\processl@denbody{\the\l@denbody\noexpand\end{\@currenvir}}%
5654   \l@denbody\@emptytoks \def\l@dbegin@stack{b}%
5655   \begingroup
5656     \expandafter\let\csname\@currenvir\endcsname\l@dcollect@@body
5657     \edef\processl@denbody{\expandafter\noexpand\csname\@currenvir\endcsname}%
5658     \processl@denbody%
5659   }%
5660 %
5661 %

```

`\l@dpush@begins` When adding a piece of the current environment's contents to `\l@denbody`, we scan it to check for additional `\begin` tokens, and add a 'b' to the stack for any that we find.

```

5662 \def\l@dpush@begins#1\begin#2{%
5663   \ifx\end#2\else b\expandafter\l@dpush@begins\fi}
5664 %
5665 %

```

`\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.

```

5666 \def\l@dcollect@@body#1\end#2{%
5667   \edef\l@dbegin@stack{\l@dpush@begins#1\begin\end
5668     \expandafter\@gobble\l@dbegin@stack}%
5669   \ifx\@empty\l@dbegin@stack
5670     \endgroup
5671     \@checkend{#2}%
5672     \addtol@denbody{#1}%
5673   \else
5674     \addtol@denbody{#1\end{#2}}%
5675   \fi
5676   \processl@denbody % A little tricky! Note the grouping
5677 }
5678 %
5679 %

```

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.

```
5680 \newcommand*{\l@dtabnoexpands}{%
5681   \let\rtab=0%
5682   \let\ctab=0%
5683   \let\ltab=0%
5684   \let\rtabtext=0%
5685   \let\ltabtext=0%
5686   \let\ctabtext=0%
5687   \let\edbeforetab=0%
5688   \let\edaftertab=0%
5689   \let\edatleft=0%
5690   \let\edatright=0%
5691   \let\edvertline=0%
5692   \let\edvertdots=0%
5693   \let\edrowfill=0%
5694 }
5695
5696 %
```

`\disable@familiarnotes` Macros to disable and restore familiar notes, to prevent them from printing multiple times in `edtabularx` and `edarrayx` environments.

`\restore@familiarnotes`

```
5697 \newcommand{\disable@familiarnotes}{%
5698   \unless\ifnofamiliar%
5699     \def\do##1{%
5700       \csletcs{footnote@@##1}{footnote##1}%
5701       \expandafter\renewcommand \csname footnote##1\endcsname[1]{%
5702         \protected@csxdef{theftnmark##1}{\csuse{thefootnote##1}}%
5703         \csuse{@footnotemark##1}%
5704       }%
5705     }%
5706     \dolistloop{\@series}%
5707   \fi%
5708 }%
5709 \newcommand{\restore@familiarnotes}{%
5710   \unless\ifnofamiliar%
5711     \def\do##1{%
5712       \csletcs{footnote##1}{footnote@@##1}%
5713     }%
5714     \dolistloop{\@series}%
5715   \fi%
```

```

5716 }%
5717
5718 %

```

`\disable@sidenotes` The same, for side notes.

`\restore@sidenotes`

```

5719 \newcommand{\disable@sidenotes}{%
5720   \let\@@ledrightnote\ledrightnote%
5721   \let\@@ledleftnote\ledleftnote%
5722   \let\@@ledsidenote\ledsidenote%
5723   \let\ledrightnote@gobble%
5724   \let\ledleftnote@gobble%
5725   \let\ledsidenote@gobble%
5726 }%
5727 \newcommand{\restore@sidenotes}{%
5728   \let\ledrightnote\@@ledrightnote%
5729   \let\ledleftnote\@@ledleftnote%
5730   \let\ledsidenote\@@ledsidenote%
5731 }%
5732 %

```

`\disable@notes` Disable/restore side and familiar notes.

`\restore@notes`

```

5733 \newcommand{\disable@notes}{%
5734   \disable@sidenotes%
5735   \disable@familiarnotes%
5736 }%
5737 \newcommand{\restore@notes}{%
5738   \restore@sidenotes%
5739   \restore@familiarnotes%
5740 }%
5741 %

```

`\EDTEXT` We need to be able to modify the `\edtext` macros and also restore their original definitions.

`\xedtext`

```

5742 \let\EDTEXT=\edtext
5743 \newcommand{\xedtext}[2]{\EDTEXT{#1}{#2}}
5744 %

```

`\EDLABEL` We need to be able to modify and restore the `\edlabel` macro.

`\xedlabel`

```

5745 \let\EDLABEL=\edlabel
5746 \newcommand*\xedlabel[1]{\EDLABEL{#1}}
5747 %

```

`\EDINDEX` Macros supporting modification and restoration of `\edindex`.

`\xedindex`

`\nulledindex`

```

5748 \let\EDINDEX=\edindex
5749 \newcommand{\xedindex}{\@bsphack%
5750 \ifnextchar [{\l@d@index}{\l@d@index[\jobname]}}
5751 \newcommand{\nulledindex}[2][\jobname]{\@bsphack\@esphack}
5752
5753 %

```

**\@line@num** Macro supporting restoration of \linenum.

```

5754 \let\@line@num=\linenum
5755 %

```

**\l@gobblearg** `\l@gobbleoptarg[⟨arg⟩]{⟨arg⟩}` replaces these two arguments (first is optional) by `\relax`.

```

5756 \newcommand*\l@gobbleoptarg}[2][\relax]%
5757
5758 %

```

**\Relax**<sub>59</sub> `\let\Relax=\relax`

**\NEXT**<sub>60</sub> `\let\NEXT=\next`

```

5761
5762 %

```

**\l@dmodforedtext** Modify and restore various macros for when \edtext is used.  
**\l@drestoreforedtext**

```

5763 \newcommand{\l@dmodforedtext}{%
5764 \let\edtext\relax
5765 \def\do##1{\global\csletcs{##1footnote}{\l@gobbleoptarg}}%
5766 \dolistloop{\@series}%
5767 \let\edindex\nulledindex
5768 \let\linenum\@gobble}
5769 \newcommand{\l@drestoreforedtext}{%
5770 \def\do##1{\global\csletcs{##1footnote}{##1@footnote}}
5771 \dolistloop{\@series}%
5772 \let\edindex\xedindex}
5773 %

```

**\l@dnullfills** Nullify and restore some column fillers, etc.  
**\l@drestorefills**

```

5774 \newcommand{\l@dnullfills}{%
5775 \def\edlabel##1{}%
5776 \def\edrowfill##1##2##3{}%
5777 }
5778 \newcommand{\l@drestorefills}{%
5779 \def\edrowfill##1##2##3{\@EDROWFILL@{##1}{##2}{##3}}%
5780 }
5781
5782 %

```

`\letsforverteilen` Gathers some lets and other code that is common to the `*verteilen*` macros.

```

5783 \newcommand{\letsforverteilen}{%
5784   \let\edtext\xedtext
5785   \let\edindex\xedindex
5786   \def\do##1{\global\csletcs{##1footnote}{##1@@footnote}}
5787   \dolistloop{\@series}%
5788   \let\linenum\@line@num
5789   \hilfe skip=\l@dcwidth%
5790   \advance\hilfe skip by -\wd\hilfe box
5791   \def\edlabel##1{\xedlabel{##1}}
5792 %
5793 %

```

`\disablel@dtabfeet` Declarations for using or using `\edtext` inside tabulars. The default at this point is for  
`\enablel@dtabfeet` `\edtext`.

```

5794 \newcommand\disablel@dtabfeet{\l@dmodforedtext}%
5795 \newcommand\enablel@dtabfeet{\l@drestoreforedtext}%
5796 %

```

### 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.

```

5797 \newcount\l@dampcount
5798 \l@dampcount=1\relax
5799 \newcount\l@dcolcount
5800 \l@dcolcount=0\relax
5801 %
5802 %

```

`\hilfe box` Some (temporary) helper items.

```

\hilfe skip
5803 \hilfe box
5804 \hilfe box
5805 \hilfe count
5806 \hilfe count
5807 %
5808 %

```

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).

```

5809 \newdimen\dcoli
5810 \newdimen\dcolii
5811 \newdimen\dcoliii
5812 \newdimen\dcoliv
5813 \newdimen\dcolv

```

```

5814 \newdimen\dc colvi
5815 \newdimen\dc colvii
5816 \newdimen\dc colviii
5817 \newdimen\dc colix
5818 \newdimen\dc colx
5819 \newdimen\dc colxi
5820 \newdimen\dc colxii
5821 \newdimen\dc colxiii
5822 \newdimen\dc colxiv
5823 \newdimen\dc colxv
5824 \newdimen\dc colxvi
5825 \newdimen\dc colxvii
5826 \newdimen\dc colxviii
5827 \newdimen\dc colxix
5828 \newdimen\dc colxx
5829 \newdimen\dc colxxi
5830 \newdimen\dc colxxii
5831 \newdimen\dc colxxiii
5832 \newdimen\dc colxxiv
5833 \newdimen\dc colxxv
5834 \newdimen\dc colxxvi
5835 \newdimen\dc colxxvii
5836 \newdimen\dc colxxviii
5837 \newdimen\dc colxxix
5838 \newdimen\dc colxxx
5839 \newdimen\dc colerr % added for error handling
5840
5841 %

```

**\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)

```

5842 \newcommand{\l@dc colwidth}{\ifcase \the\l@dc colcount \dcoli %???
5843 \or \dcoli \or \dc colii \or \dc coliii
5844 \or \dc coliv \or \dc colv \or \dc colvi
5845 \or \dc colvii \or \dc colviii \or \dc colix \or \dc colx
5846 \or \dc colxi \or \dc colxii \or \dc colxiii
5847 \or \dc colxiv \or \dc colxv \or \dc colxvi
5848 \or \dc colxvii \or \dc colxviii \or \dc colxix \or \dc colxx
5849 \or \dc colxxi \or \dc colxxii \or \dc colxxiii
5850 \or \dc colxxiv \or \dc colxxv \or \dc colxxvi
5851 \or \dc colxxvii \or \dc colxxviii \or \dc colxxix \or \dc colxxx
5852 \else \dc colerr \fi}
5853
5854 %

```

**\step1@dc colcount** This increments the column counter, and issues an error message if it is too large.

```

5855 \newcommand*{\step1@dc colcount}{\advance\l@dc colcount\@ne
5856 \ifnum\l@dc colcount>30\relax

```

```

5857 \led@err@TooManyColumns
5858 \fi}
5859
5860 %

```

**\l@setmaxcolwidth** Sets the column width to the maximum value seen so far.

```

5861 \newcommand{\l@setmaxcolwidth}{%
5862   \ifdim\l@dcwidth < \wd\hilfsbox
5863     \l@dcwidth = \wd\hilfsbox
5864   \else \relax \fi}
5865
5866 %

```

**\measurecell** Measure (recursively) the width required for a math cell.

```

5867 \def\measurecell #1{%
5868   \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
5869     \else\l@dcwidth=0%
5870     \l@dcwidth=0%
5871     \let\NEXT\measurecell%
5872   \fi%
5873   \else\setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5874     \step\l@dcwidth%
5875     \l@setmaxcolwidth%
5876     \let\NEXT\measurecell%
5877   \fi\NEXT}
5878
5879 %

```

**\measuretextcell** Measure (recursively) the width required for a text cell.

```

5880 \def\measuretextcell #1{%
5881   \ifx #1\ \ifnum\l@dcwidth=0\let\NEXT\relax%
5882     \else\l@dcwidth=0%
5883     \l@dcwidth=0%
5884     \let\NEXT\measuretextcell%
5885   \fi%
5886   \else\setbox\hilfsbox=\hbox{#1}%
5887     \step\l@dcwidth%
5888     \l@setmaxcolwidth%
5889     \let\NEXT\measuretextcell%
5890   \fi\NEXT}
5891
5892 %

```

**\measuremrow** Measure (recursively) the width required for a math row.

```

5893 \def\measuremrow #1{%
5894   \ifx #1\ \let\NEXT\relax%

```

```

5895 \else\measuremcell #1&\\&\\&%
5896 \let\NEXT\measuremrow%
5897 \fi\NEXT}
5898 %

```

**\measuretrow** Measure (recursively) the width required for a text row.

```

5899 \def\measuretrow #1\\{%
5900 \ifx #1&\let\NEXT\relax%
5901 \else\measuretcell #1&\\&\\&%
5902 \let\NEXT\measuretrow%
5903 \fi\NEXT}
5904
5905 %

```

**\edtabcolsep** The length `\edtabcolsep` controls the distance between columns.

```

5906 \newskip\edtabcolsep
5907 \global\edtabcolsep=10pt
5908
5909 %

```

**\variab**<sub>10</sub> `\newcommand{\variab}{\relax}`

```

5911
5912 %

```

**\l@dcheckcols** Check that the number of columns is consistent.

```

5913 \newcommand*{\l@dcheckcols}{%
5914 \ifnum\l@dcolcount=1\relax
5915 \else
5916 \ifnum\l@dampcount=1\relax
5917 \else
5918 \ifnum\l@dcolcount=\l@dampcount\relax
5919 \else
5920 \l@d@err@UnequalColumns
5921 \fi
5922 \fi
5923 \l@dampcount=\l@dcolcount
5924 \fi}
5925
5926 %

```

**\edfilldimen** A length.

```

5927 \newdimen\edfilldimen
5928 \edfilldimen=0pt
5929
5930 %

```



`\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`.

```

5931 \newcounter{addcolcount}
5932 \renewcommand{\theadcolcount}{\roman{addcolcount}}
5933 %

```

### XXVIII.2.3 Tabular typesetting

`\setmcellright` Typeset (recursively) cells of display math right justified.

```

5934 \def\setmcellright #1{\def\edlabel##1{}}%
5935 \let\edindex\nulledindex
5936 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
5937 \let\Next\relax%
5938 \else\l@dcolcount=0%
5939 \let\Next=\setmcellright%
5940 \fi%
5941 \else%
5942 \disablel@dtabfeet%
5943 \step1@dcolcount%
5944 \disable@notes%
5945 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5946 \restore@notes%
5947 \letsforverteilen%
5948 \hskip\hilfsskip$\displaystyle{#1}$%
5949 \hskip\edtabcolsep%
5950 \let\Next=\setmcellright%
5951 \fi\Next}
5952 %
5953 %

```

`\settcellright` Typeset (recursively) cells of text right justified.

```

5954 \def\settcellright #1{\def\edlabel##1{}}%
5955 \let\edindex\nulledindex
5956 \ifx #1\ \ifnum\l@dcolcount=0%\removelastskip
5957 \let\Next\relax%
5958 \else\l@dcolcount=0%
5959 \let\Next=\settcellright%
5960 \fi%
5961 \else%
5962 \disablel@dtabfeet%
5963 \step1@dcolcount%
5964 \disable@notes%
5965 \setbox\hilfsbox=\hbox{#1}%
5966 \restore@notes%
5967 \letsforverteilen%
5968 \hskip\hilfsskip#1%
5969 \hskip\edtabcolsep%

```

```

5970         \let\Next=\settccllright%
5971         \fi\Next}
5972 %

```

**\setmcellleft** Typeset (recursively) cells of display math left justified.

```

5973 \def\setmcellleft #1&{\def\edlabel##1{}%
5974     \let\edindex\nulledindex
5975     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
5976         \else\l@dcolcount=0%
5977         \let\Next=\setmcellleft%
5978         \fi%
5979     \else \disablel@dtabfeet%
5980         \step1@dcolcount%
5981         \disable@notes%
5982         \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
5983         \restore@notes%
5984         \letsforverteilen%
5985         $\displaystyle{#1}$\hskip\hilfsskip\hskip\edtabcolsep%
5986         \let\Next=\setmcellleft%
5987     \fi\Next}
5988
5989 %

```

**\settcclleft** Typeset (recursively) cells of text left justified.

```

5990 \def\settcclleft #1&{\def\edlabel##1{}%
5991     \let\edindex\nulledindex
5992     \ifx #1\\ \ifnum\l@dcolcount=0 \let\Next\relax%
5993         \else\l@dcolcount=0%
5994         \let\Next=\settcclleft%
5995         \fi%
5996     \else \disablel@dtabfeet%
5997         \step1@dcolcount%
5998         \disable@notes%
5999         \setbox\hilfsbox=\hbox{#1}%
6000         \restore@notes%
6001         \letsforverteilen%
6002         #1\hskip\hilfsskip\hskip\edtabcolsep%
6003         \let\Next=\settcclleft%
6004     \fi\Next}
6005 %

```

**\setmcellcenter** Typeset (recursively) cells of display math centered.

```

6006 \def\setmcellcenter #1&{\def\edlabel##1{}%
6007     \let\edindex\nulledindex
6008     \ifx #1\\ \ifnum\l@dcolcount=0\let\Next\relax%
6009         \else\l@dcolcount=0%
6010         \let\Next=\setmcellcenter%

```

```

6011 \fi%
6012 \else \disablel@dtabfeet%
6013 \stepl@dcolcount%
6014 \disable@notes%
6015 \setbox\hilfsbox=\hbox{$\displaystyle{#1}$}%
6016 \restore@notes%
6017 \letsforverteilen%
6018 \hskip 0.5\hilfsskip$\displaystyle{#1}$\hskip0.5\hilfsskip%
6019 \hskip\edtabcolsep%
6020 \let\Next=\setmcellcenter%
6021 \fi\Next}
6022
6023 %

```

**\settcclcenter** Typeset (recursively) cells of text centered.

```

6024 \def\settcclcenter #1{\def\edlabel##1{%
6025 \let\edindex\nulledindex
6026 \ifx #1\ \ifnum\l@dcolcount=0 \let\Next\relax%
6027 \else\l@dcolcount=0%
6028 \let\Next=\settcclcenter%
6029 \fi%
6030 \else \disablel@dtabfeet%
6031 \stepl@dcolcount%
6032 \disable@notes%
6033 \setbox\hilfsbox=\hbox{#1}%
6034 \restore@notes%
6035 \letsforverteilen%
6036 \hskip 0.5\hilfsskip #1\hskip 0.5\hilfsskip%
6037 \hskip\edtabcolsep%
6038 \let\Next=\settcclcenter%
6039 \fi\Next}
6040
6041 %

```

**\NEXT**<sub>142</sub> \let\NEXT=\relax

```

6043
6044 %

```

**\setmrowright** Typeset (recursively) rows of right justified math.

```

6045 \def\setmrowright #1\{%
6046 \ifx #1& \let\NEXT\relax
6047 \else \centerline{\setmcellright #1&\&\&}
6048 \let\NEXT=\setmrowright
6049 \fi\NEXT}
6050 %

```

**\settroright** Typeset (recursively) rows of right justified text.

```

6051 \def\settroright #1\{\%
6052   \ifx #1& \let\NEXT\relax
6053   \else \centerline{\settcellright #1&\&\&\&}
6054   \let\NEXT=\settroright
6055   \fi\NEXT}
6056
6057 %

```

**\setmrowleft** Typeset (recursively) rows of left justified math.

```

6058 \def\setmrowleft #1\{\%
6059   \ifx #1& \let\NEXT\relax
6060   \else \centerline{\setmcellleft #1&\&\&\&}
6061   \let\NEXT=\setmrowleft
6062   \fi\NEXT}
6063 %

```

**\settrorleft** Typeset (recursively) rows of left justified text.

```

6064 \def\settrorleft #1\{\%
6065   \ifx #1& \let\NEXT\relax
6066   \else \centerline{\settcellleft #1&\&\&\&}
6067   \let\NEXT=\settrorleft
6068   \fi\NEXT}
6069
6070 %

```

**\setmrowcenter** Typeset (recursively) rows of centered math.

```

6071 \def\setmrowcenter #1\{\%
6072   \ifx #1& \let\NEXT\relax%
6073   \else \centerline{\setmcellcenter #1&\&\&\&}
6074   \let\NEXT=\setmrowcenter
6075   \fi\NEXT}
6076 %

```

**\settrorcenter** Typeset (recursively) rows of centered text.

```

6077 \def\settrorcenter #1\{\%
6078   \ifx #1& \let\NEXT\relax
6079   \else \centerline{\settcellcenter #1&\&\&\&}
6080   \let\NEXT=\settrorcenter
6081   \fi\NEXT}
6082
6083 %

```

```

6084 \newcommand{\nullsetzen}{\%
6085   \stepl@dc@colcount%

```

```

6086 \l@dcwidth=0pt%
6087 \ifnum\l@dcwidth=30\let\NEXT\relax%
6088 \l@dcwidth=0\relax
6089 \else\let\NEXT\relax%
6090 \fi\NEXT}
6091
6092 %

```

**\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.

```

6093 \newcommand{\edatleft}[3][\@empty]{%
6094 \ifx#1\@empty
6095 \vbox to 10pt{\vss\hbox{$\left#2\vrule width0pt height #3
6096 \depth 0pt \right. $\hss}\vfil}
6097 \else
6098 \vbox to 4pt{\vss\hbox{$#1\left#2\vrule width0pt height #3
6099 \depth 0pt \right. $\}\vfil}
6100 \fi}
6101 %

```

**\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.

```

6102 \newcommand{\edatright}[3][\@empty]{%
6103 \ifx#1\@empty
6104 \vbox to 10pt{\vss\hbox{$\left.\vrule width0pt height #3
6105 \depth 0pt \right#2 $\hss}\vfil}
6106 \else
6107 \vbox to 4pt{\vss\hbox{$\left.\vrule width0pt height #3
6108 \depth 0pt \right#2 #1 $\}\vfil}
6109 \fi}
6110 %
6111 %

```

**\edvertline** `\edvertline{ $\langle len \rangle$ }` vertical line  $\langle len \rangle$  high.

```

6112 \newcommand{\edvertline}[1]{\vbox to 8pt{\vss\hbox{\vrule height #1}\vfil}}
6113
6114 %

```

**\edvertdots** `\edvertdots{ $\langle len \rangle$ }` vertical dotted line  $\langle len \rangle$  high.

```

6115 \newcommand{\edvertdots}[1]{\vbox to 1pt{\vss\vbox to #1%
6116 {\cleaders\hbox{$\mathhbox{.}\vbox to 0.5em{ }\vfil}}}}
6117
6118 %

```

**\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  $\text{\TeX}$  style reimplementation of the original `\@add@`.

```

6119 \newcommand{\l@dtabaddcols}[2]{%
6120   \l@dccheckstartend{#1}{#2}%
6121   \ifl@dstartendok
6122     \setcounter{addcolcount}{#1}%
6123     \@whilenum \value{addcolcount}<#2\relax \do
6124       {\advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6125        \advance\edfilldimen by \edtabcolsep
6126        \stepcounter{addcolcount}}%
6127     \advance\edfilldimen by \the \csname dcol\theaddcolcount\endcsname
6128     \fi
6129   }
6130
6131   %

```

`\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.

```

6132 \newif\ifl@dstartendok
6133 \newcommand{\l@dccheckstartend}[2]{%
6134   \l@dstartendoktrue
6135   \ifnum #1<\@ne
6136     \l@dstartendokfalse
6137     \led@err@LowStartColumn
6138   \fi
6139   \ifnum #2>30\relax
6140     \l@dstartendokfalse
6141     \led@err@HighEndColumn
6142   \fi
6143   \ifnum #1>#2\relax
6144     \l@dstartendokfalse
6145     \led@err@ReverseColumns
6146   \fi
6147 }
6148
6149 %

```

`\edrowfill` `\edrowfill{<startcol>}{<endcol>}` fill fills columns `<startcol>` to `<endcol>` inclusive with `<fill>` (e.g. `\hrulefill`, `\upbracefill`). This is a  $\text{\TeX}$  style reimplementation and generalization of the original `\waklam`, `\Waklam`, `\waklamec`, `\wastricht` and `\wapunktel` macros.

```

6150 \newcommand*\edrowfill}[3]{%
6151   \l@dtabaddcols{#1}{#2}%
6152   \hb@xt@ \the\l@dcolwidth{\hb@xt@ \the\edfilldimen{#3}\hss}}
6153 \let\@edrowfill@=\edrowfill
6154 \def\@EDROWFILL@#1#2#3{\@edrowfill@{#1}{#2}{#3}}
6155
6156 %

```

`\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`.

```
6157 \newcommand{\leftltab}[1]{%
6158   \hb@xt@z@{\vbox{\edtabindent%
6159     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6160
6161 %
```

`\leftrtab`     `\leftrtab{⟨text⟩}{⟨math⟩}` for `\edbeforetab` in `\rtab`.

```
6162 \newcommand{\leftrtab}[2]{%
6163   #2\hb@xt@z@{\vbox{\edtabindent%
6164     \advance\Hilfsskip by\dcoli%
6165     \moveleft\Hilfsskip\hbox{\ #1}}\hss}}
6166
6167 %
```

`\leftctab`     `\leftctab{⟨text⟩}{⟨math⟩}` for `\edbeforetab` in `\ctab`.

```
6168 \newcommand{\leftctab}[2]{%
6169   \hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6170     \advance\Hilfsskip by 0.5\dcoli%
6171     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6172     \disablel@dtabfeet$\displaystyle{#2}$}%
6173     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6174     \moveleft\Hilfsskip\hbox{\ #1}}\hss}%
6175   #2}
6176
6177 %
```

`\rightctab`     `\rightctab{⟨math⟩}{⟨text⟩}` for `\edaftertab` in `\ctab`.

```
6178 \newcommand{\rightctab}[2]{%
6179   \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6180   \disablel@dtabfeet#2\l@dampcount=\l@dcolcount%
6181   #1\hb@xt@z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6182     \advance\Hilfsskip by 0.5\l@dcolwidth%
6183     \advance\Hilfsskip by -\wd\hilfsbox%
6184     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6185     \disablel@dtabfeet$\displaystyle{#1}$}%
6186     \advance\Hilfsskip by -0.5\wd\hilfsbox%
6187     \advance\Hilfsskip by \edtabcolsep%
6188     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6189   }
6190
6191 %
```

`\rightltab` `\rightltab{ $\langle math \rangle}{\langle text \rangle}$`  for `\edaftertab` in `\ltab`.

```

6192 \newcommand{\rightltab}[2]{%
6193     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6194     \disablel@dtabfeet#2}\l@dampcount=\l@dcolcount%
6195     #1\hb@xt@{\z@{\vbox{\edtabindent\l@dcolcount=\l@dampcount%
6196     \advance\Hilfsskip by\l@dcolwidth%
6197     \advance\Hilfsskip by-\wd\hilfsbox%
6198     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6199     \disablel@dtabfeet$\displaystyle{#1}$}%
6200     \advance\Hilfsskip by-\wd\hilfsbox%
6201     \advance\Hilfsskip by\edtabcolsep%
6202     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6203 }
6204
6205 %

```

`\rightrtab` `\rightrtab{ $\langle math \rangle}{\langle text \rangle}$`  for `\edaftertab` in `\rtab`.

```

6206 \newcommand{\rightrtab}[2]{%
6207     \setbox\hilfsbox=\hbox{\def\edlabel##1{}}%
6208     \disablel@dtabfeet#2}%
6209     #1\hb@xt@{\z@{\vbox{\edtabindent%
6210     \advance\Hilfsskip by-\wd\hilfsbox%
6211     \advance\Hilfsskip by\edtabcolsep%
6212     \moveright\Hilfsskip\hbox{ #2}}\hss}%
6213 }
6214
6215 %

```

`\rtab` `\rtab{ $\langle body \rangle}$`  typesets  $\langle body \rangle$  as an array with the entries right justified.

`\edbeforetab` The process is first to measure the  $\langle body \rangle$  to get the column widths, and then in a  
`\edaftertab` second pass to typeset the body.

```

6216 \newcommand{\rtab}[1]{%
6217     \l@dnnullfills
6218     \def\edbeforetab##1##2{\lefttab{##1}{##2}}%
6219     \def\edaftertab##1##2{\righttab{##1}{##2}}%
6220     \measurebody{#1}%
6221     \l@dretofills
6222     \variab
6223     \setmrowright #1\&\&%
6224     \enablel@dtabfeet}
6225
6226 %

```

`\measurebody` `\measurebody{ $\langle body \rangle}$`  measures the array  $\langle body \rangle$ .

```

6227 \newcommand{\measurebody}[1]{%
6228     \disablel@dtabfeet%

```



```

6229 \l@dcolcount=0%
6230 \nullsetzen%
6231 \l@dcolcount=0
6232 \measuremrow #1\\&\\%
6233 \global\l@dampcount=1}
6234
6235 %

```

**\rtabtext** `\rtabtext{<body>}` typesets <body> as a tabular with the entries right justified.

```

6236 \newcommand{\rtabtext}[1]{%
6237 \l@dnnullfills
6238 \measuretbody{#1}%
6239 \l@drestorefills
6240 \variab
6241 \settroright #1\\&\\%
6242 \enablel@dtabfeet}
6243
6244 %

```

**\measuretbody** `\measuretbody{<body>}` measures the tabular <body>.

```

6245 \newcommand{\measuretbody}[1]{%
6246 \disable@notes%
6247 \disablel@dtabfeet%
6248 \l@dcolcount=0%
6249 \nullsetzen%
6250 \l@dcolcount=0
6251 \measuretror #1\\&\\%
6252 \restore@notes%
6253 \global\l@dampcount=1}
6254
6255 %

```

**\ltab** Array with entries left justified.

```

\edbeforetab
\edaftertab
6256 \newcommand{\ltab}[1]{%
6257 \l@dnnullfills
6258 \def\edbeforetab##1##2{\leftltab{##1}{##2}}%
6259 \def\edaftertab##1##2{\rightltab{##1}{##2}}%
6260 \measuretbody{#1}%
6261 \l@drestorefills
6262 \variab
6263 \setmrowleft #1\\&\\%
6264 \enablel@dtabfeet}
6265
6266 %

```

**\ltabtext** Tabular with entries left justified.

```

6267 \newcommand{\ltabtext}[1]{%
6268   \l@dnnullfills
6269   \measuretbody{#1}%
6270   \l@drestorefills
6271   \variab
6272   \settrorleft #1\\&\\%
6273   \enablel@dtabfeet}
6274
6275 %

```

**\ctab** Array with centered entries.

```

\edbeforetab
\edaftertab
6276 \newcommand{\ctab}[1]{%
6277   \l@dnnullfills
6278   \def\edbeforetab##1##2{\leftctab{##1}{##2}}%
6279   \def\edaftertab##1##2{\rightctab{##1}{##2}}%
6280   \measuretbody{#1}%
6281   \l@drestorefills
6282   \variab
6283   \setmrowcenter #1\\&\\%
6284   \enablel@dtabfeet}
6285
6286 %

```

**\ctabtext** Tabular with entries centered.

```

6287 \newcommand{\ctabtext}[1]{%
6288   \l@dnnullfills
6289   \measuretbody{#1}%
6290   \l@drestorefills
6291   \variab
6292   \settrorcenter #1\\&\\%
6293   \enablel@dtabfeet}
6294
6295 %

```

```

\spreadtext 6296 \newcommand{\spreadtext}[1]{%\l@dcolcount=\l@dampcount%
6297   \hb@xt@ \the\l@dcolwidth{\hbox{#1}\hss}}
6298 %

```

```

\spreadmath 6299 \newcommand{\spreadmath}[1]{%
6300   \hb@xt@ \the\l@dcolwidth{\hbox{$\displaystyle{#1}$}\hss}}
6301
6302 %

```

**\HILFSskip** More helpers.

**\Hilfsskip**

```

6303 \newskip\HILFSskip
6304 \newskip\Hilfsskip
6305
6306 %

```

```

\EDTABINDENT \newcommand{\EDTABINDENT}{%
6307
6308   \ifnum\l@dcolcount=30\let\NEXT\relax\l@dcolcount=0%
6309   \else\step1@dcolcount%
6310     \advance\Hilfsskip by\l@dcolwidth%
6311     \ifdim\l@dcolwidth=0pt\advance\hilfscount\@ne
6312     \else\advance\Hilfsskip by \the\hilfscount\edtabcolsep%
6313     \hilfscount=1\fi%
6314     \let\NEXT=\EDTABINDENT%
6315   \fi\NEXT}%
6316 %

```

`\edtabindent` (was `\tabindent`)

```

6317 \newcommand{\edtabindent}{%
6318   \l@dcolcount=0\relax
6319   \Hilfsskip=0pt%
6320   \hilfscount=1\relax
6321   \EDTABINDENT%
6322   \hilfsskip=\hsize%
6323   \advance\hilfsskip -\Hilfsskip%
6324   \Hilfsskip=0.5\hilfsskip%
6325   }%
6326
6327 %

```

`\EDTAB` (was `\TAB`)

```

6328 \def\EDTAB #1|#2|{%
6329   \setbox\tabhilfbox=\hbox{$\displaystyle{#1}$}%
6330   \setbox\tabHilfbox=\hbox{$\displaystyle{#2}$}%
6331   \advance\tabelskip -\wd\tabhilfbox%
6332   \advance\tabelskip -\wd\tabHilfbox%
6333   \unhbox\tabhilfbox\hskip\tabelskip%
6334   \unhbox\tabHilfbox}%
6335
6336 %

```

`\EDTABtext` (was `\TABtext`)

```

6337 \def\EDTABtext #1|#2|{%
6338   \setbox\tabhilfbox=\hbox{#1}%
6339   \setbox\tabHilfbox=\hbox{#2}%
6340   \advance\tabelskip -\wd\tabhilfbox%
6341   \advance\tabelskip -\wd\tabHilfbox%

```



```

6371 }
6372 \renewcommand{\endquotation}{\par%
6373     \global\leftskip=0pt%
6374     \global\rightskip=0pt%
6375     \leavevmode%
6376     \skipnumbering%
6377     \ifautopar%
6378         \vskip-\parskip%
6379     \else%
6380         \vskip\topsep%
6381     \fi%
6382 }
6383 \renewcommand{\quote}{\par\leavevmode%
6384     \parindent=0pt%
6385     \skipnumbering%
6386     \ifautopar%
6387         \vskip-\parskip%
6388     \else%
6389         \vskip\topsep%
6390     \fi%
6391     \global\leftskip=\leftmargin%
6392     \global\rightskip=\leftmargin%
6393 }
6394 \renewcommand{\endquote}{\par%
6395     \global\leftskip=0pt%
6396     \global\rightskip=0pt%
6397     \leavevmode%
6398     \skipnumbering%
6399     \ifautopar%
6400         \vskip-\parskip%
6401     \else%
6402         \vskip\topsep%
6403     \fi%
6404 }
6405 \fi
6406 }
6407 %

```

## XXX Section's title commands

### XXX.1 Commands to disable some feature

`\ledsectnotoc` The `\ledsectnotoc` only disables the `\addcontentsline` macro.

```

6408 \newcommand{\ledsectnotoc}{\let\addcontentsline\@gobblethree}
6409 %

```

`\ledsectnomark` The `\ledsectnomark` only disables the `\chaptermark`, `\sectionmark` and `\subsectionmark` macros.

```

6410 \newcommand{\ledsectnomark}{%
6411   \let\chaptermark@gobble%
6412   \let\sectionmark@gobble%
6413   \let\subsectionmark@gobble%
6414 }
6415 %

```

## 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  $\TeX$  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  $\TeX$  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).

```

6416 \catcode`\#=12
6417 \notbool{@noeled@sec}{%
6418 %

```

`\beforeeledchapter` For technical reasons, not yet solved, page-breaking before chapters can't be made automatically by `eledmac`. Users have to use `\beforeeledchapter`.

```

6419 \ifl@dmemoir
6420   \newcommand\beforeeledchapter{%
6421     \clearforchapter%
6422   }
6423 \else
6424   \newcommand\beforeeledchapter{%
6425     \if@openright%
6426       \cleardoublepage%
6427     \else%
6428       \clearpage%
6429     \fi%
6430   }
6431 \fi
6432 %

```

### 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.

```

6433 \newif\if@eled@sectioning
6434 %

```

`\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.

```

6435 \def\print@rightmargin@eledsection{%
6436   \if@eled@sectioning%
6437     \begingroup%
6438     \if@RTL%
6439       \let\llap\rlap%
6440       \let\leftlinenum\rightlinenum%
6441       \let\leftlinenumR\rightlinenumR%
6442       \let\l@drd@ta\l@dld@ta%
6443       \let\l@drsn@te\l@dlsn@te%
6444     \fi%
6445     \hfill\l@drd@ta \csuse{LR}{\l@drsn@te}%
6446   \endgroup%
6447 \fi%
6448 }%
6449
6450 \def\print@leftmargin@eledsection{%
6451   \if@eled@sectioning%
6452     \leavevmode%
6453     \begingroup%
6454     \if@RTL%
6455       \let\rlap\llap%

```

```

6456 \let\rightlinenum\leftlinenum%
6457 \let\rightlinenumR\leftlinenumR%
6458 \let\l@dld@ta\l@dld@ta%
6459 \let\l@dlsn@te\l@dlsn@te%
6460 \fi%
6461 \l@dld@ta\csuse{LR}{\l@dlsn@te}%
6462 \endgroup%
6463 \fi%
6464 }%
6465
6466 %

```

### XXX.5 Patching standard commands

`\chapter` We have to patch L<sup>A</sup>T<sub>E</sub>X, book and memoir sectioning commands in order to:

`\M@sect`

`\@mem@old@ssect`

`\@makechapterhead`

`\@makechapterhead`

`\@makeschapterhead`

`\@sect`

`\@ssect`

- Disable `\edtext` inside.
- Disable page breaking (for `\chapter`).
- Add line numbers and sidenotes.

Unfortunately, Maïeul Rouquette was not able to try if memoir is loaded. That is why `eledmac` tries to define for both standard class and memoir class.

```

6467 \AtBeginDocument{%
6468 \patchcmd{\chapter}{\clearforchapter}{%
6469 \if@eled@sectioning\else%
6470 \ifl@dprintingpages\else%
6471 \clearforchapter%
6472 \fi%
6473 \fi%
6474 }
6475 {}
6476 {}
6477
6478
6479 \pretocmd{\M@sect}
6480 {\let\old@edtext=\edtext%
6481 \let\edtext=\dummy@edtext@showlemma%
6482 }
6483 {}
6484 {}
6485
6486 \apptocmd{\M@sect}
6487 {\let\edtext=\old@edtext}
6488 {}
6489 {}
6490
6491 \patchcmd{\M@sect}

```



```

6492 { #9}
6493 { #9%
6494 \print@rightmargin@eledsection%
6495 }
6496 {}
6497 {}
6498
6499 \patchcmd{\M@sect}
6500 {\hskip #3\relax}
6501 {\hskip #3\relax%
6502 \print@leftmargin@eledsection%
6503 }
6504 {}
6505 {}
6506
6507 \patchcmd{\@mem@old@ssect}
6508 {#5}
6509 {#5%
6510 \print@leftmargin@eledsection%
6511 }
6512 {}
6513 {}
6514
6515 \patchcmd{\@mem@old@ssect}
6516 {\hskip #1}
6517 {\hskip #1%
6518 \print@rightmargin@eledsection%
6519 }
6520 {}
6521 {}
6522
6523 \patchcmd{\chapter}{\if@openright\cleardoublepage\else\clearpage\fi}{%
6524 \if@eled@sectioning\else%
6525 \ifl@dprintingpages\else%
6526 \if@openright\cleardoublepage\else\clearpage\fi%No clearpage inside a
\Pages: will keep critical notes from printing on the title page. Here for
classical classes
6527 \fi%
6528 \fi%
6529 }%
6530 {}%
6531 {}%
6532
6533 \patchcmd{\scr@startchapter}{\if@openright\cleardoublepage\else\clearpage\
fi}{%
6534 \if@eled@sectioning\else%
6535 \ifl@dprintingpages\else%
6536 \if@openright\cleardoublepage\else\clearpage\fi%No clearpage inside a
\Pages: will keep critical notes from printing on the title page. Here for
scrbook.

```

```

6537 \fi%
6538 \fi%
6539 }
6540 {}
6541 {}
6542
6543 \patchcmd{\@makechapterhead}
6544 {#1}
6545 {\print@leftmargin@eledsection%
6546 #1%
6547 \print@rightmargin@eledsection%
6548 }
6549 {}
6550 {}
6551
6552 \patchcmd{\@makechapterhead}% For BIDI
6553 {\if@RTL\raggedleft\else\raggedright\fi}%
6554 {\if@eled@sectioning\else%
6555 \if@RTL\raggedleft\else\raggedright\fi%
6556 \fi%
6557 }%
6558 {}%
6559 {}%
6560
6561 \patchcmd{\@makeschapterhead}
6562 {#1}
6563 {\print@leftmargin@eledsection%
6564 #1%
6565 \print@rightmargin@eledsection%
6566 }
6567 {}
6568 {}
6569
6570 \pretocmd{\@sect}
6571 {\let\old@edtext=\edtext
6572 \let\edtext=\dummy@edtext@showlemma%
6573 }
6574 {}
6575 {}
6576
6577 \apptocmd{\@sect}
6578 {\let\edtext=\old@edtext}
6579 {}
6580 {}
6581
6582 \pretocmd{\@ssect}
6583 {\let\old@edtext=\edtext%
6584 \let\edtext=\dummy@edtext@showlemma%
6585 }
6586 {}

```

```

6587 {}
6588
6589 \apptocmd{\@ssect}
6590 {\let\edtext=\old@edtext}
6591 {}
6592 {}
6593
6594 %

```

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.

```

6595 \@ifpackageloaded{nameref}{
6596
6597   \patchcmd{\NR@sect}
6598     {#8}
6599     {#8%
6600       \print@rightmargin@eledsection%
6601     }
6602     {}
6603     {}
6604
6605   \patchcmd{\NR@ssect}
6606     {\hskip #3\relax}
6607     {\hskip #3\relax%
6608       \print@leftmargin@eledsection%
6609     }
6610     {}
6611     {}
6612
6613   \patchcmd{\NR@ssect}
6614     {#5}
6615     {#5%
6616       \print@rightmargin@eledsection%
6617     }
6618     {}
6619     {}
6620
6621   \patchcmd{\NR@ssect}
6622     {\hskip #1}
6623     {\hskip #1%
6624       \print@leftmargin@eledsection%
6625     }
6626     {}
6627     {}
6628   }%
6629   {
6630     \patchcmd{\@sect}
6631       {#8}
6632       {#8%
6633         \print@rightmargin@eledsection%

```

```

6634 }
6635 {}
6636 {}
6637
6638 \patchcmd{\@sect}
6639   {\hskip #3\relax}
6640   {\hskip #3\relax%
6641   \print@leftmargin@eledsection%
6642   }
6643   {}
6644   {}
6645
6646 \patchcmd{\@ssect}
6647   {#5}
6648   {#5%
6649   \print@rightmargin@eledsection%
6650   }
6651   {}
6652   {}
6653
6654 \patchcmd{\@ssect}
6655   {\hskip #1}
6656   {\hskip #1%
6657   \print@leftmargin@eledsection%
6658   }
6659   {}
6660   {}
6661 }%
6662 }
6663 %

```

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.

```

6664 {}}%
6665 \protect\catcode`\#=6 %Space NEEDS by \catcode
6666 \notbool{@noeled@sec}{%
6667 %

```

### 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.

```

6668 \newwrite\eled@sectioning@out
6669 %

```

`\eledchapter` And now, the user sectioning commands, which write to the file, and also add content as a “normal” line.

`\eledsection`

`\eledsubsection`

`\eledsubsubsection`

`\eledchapter*`

`\eledsection*`

`\eledsubsection*`

`\eledsubsubsection*`

```

6670 \newcommand{\eledchapter}[2] [] {%
6671   #2%
6672   \ifledRcol%
6673     \immediate\write\eled@sectioningR@out{%
6674       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6675     }%
6676   \else%
6677     \immediate\write\eled@sectioning@out{%
6678       \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6679     }%
6680   \fi%
6681 }
6682
6683 \newcommand{\eledsection}[2] [] {%
6684   #2%
6685   \ifledRcol%
6686     \immediate\write\eled@sectioningR@out{%
6687       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6688     }%
6689   \else%
6690     \immediate\write\eled@sectioning@out{%
6691       \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6692     }%
6693   \fi%
6694 }
6695
6696 \newcommand{\eledsubsection}[2] [] {%
6697   #2%
6698   \ifledRcol%
6699     \immediate\write\eled@sectioningR@out{%
6700       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{-}{R}
6701     }%
6702   \else%
6703     \immediate\write\eled@sectioning@out{%
6704       \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{-}{-}
6705     }%
6706   \fi%
6707 }
6708 \newcommand{\eledsubsubsection}[2] [] {%
6709   #2%
6710   \ifledRcol%
6711     \immediate\write\eled@sectioningR@out{%
6712       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}
6713     }-}{R}
6714   \else%
6715     \immediate\write\eled@sectioning@out{%
6716       \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}
6717     }-}{-}

```

```

6717     }%
6718     \fi%
6719 }
6720
6721
6722 \WithSuffix\newcommand\eledchapter*[2] [] {%
6723     #2%
6724     \ifledRcol%
6725         \immediate\write\eled@sectioningR@out{%
6726             \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6727         }%
6728     \else%
6729         \immediate\write\eled@sectioning@out{%
6730             \string\eled@chapter{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
6731         }%
6732     \fi%
6733 }
6734
6735 \WithSuffix\newcommand\eledsection*[2] [] {%
6736     #2%
6737     \ifledRcol%
6738         \immediate\write\eled@sectioningR@out{%
6739             \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6740         }%
6741     \else%
6742         \immediate\write\eled@sectioning@out{%
6743             \string\eled@section{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}{*}{L}
6744         }%
6745     \fi%
6746 }
6747
6748 \WithSuffix\newcommand\eledsubsection*[2] [] {%
6749     #2%
6750     \ifledRcol%
6751         \immediate\write\eled@sectioningR@out{%
6752             \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsR}{*}{R}
6753         }%
6754     \else%
6755         \immediate\write\eled@sectioning@out{%
6756             \string\eled@subsection{#1}{\unexpanded{#2}}{\the\l@dnumpstartsL}
6757             }{*}{L}
6758         }%
6759     \fi%
6760 }
6761
6762 \WithSuffix\newcommand\eledsubsubsection*[2] [] {%
6763     #2%
6764     \ifledRcol%
6765         \immediate\write\eled@sectioningR@out{%

```

```

6765 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsR
6766 }{*}{R}
6767 }%
6768 \else%
6769 \immediate\write\eled@sectioning@out{%
6770 \string\eled@subsubsection{#1}{\unexpanded{#2}}{\the\l@dumpstartsL
6771 }{*}{L}
6772 }%
6773 \fi%
6774 }
6775 %

```

### 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.

```

6774 \def\eled@chapter#1#2#3#4#5{%
6775 \ifstrempy{#4}%
6776 {%
6777 \ifstrempy{#1}%
6778 {%
6779 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter{#2}}%
6780 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{\
chaptermark{#2}}}%
6781 }%Need for \pairs, because of using parbox.
6782 {%
6783 \global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter[#1]{#2}}%
6784 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext{\
chaptermark{#2}}}%Need for \pairs, because of using parbox.
6785 }%
6786 }%
6787 {%
6788 \ifstrempy{#1}%
6789 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#2}}}%
6790 {\global\csdef{eled@sectioning@#3#5}{\let\edtext=\
dummy@edtext@showlemma\chapter*{#1}{#2}}}%Bug in LaTeX!
6791 }%

```

```

6792 \listcsgadd{eled@sections#5@@}{#3}%
6793 }
6794 \def\eled@section#1#2#3#4#5{%
6795 \ifstrempy{#4}%
6796 {\ifstrempy{#1}%
6797 {%
6798 \global\csdef{eled@sectioning@#3#5}{\section{#2}}%
6799 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\
sectionmark{#2}}%Need for \pairs, because of using parbox.
6800 }%
6801 {%
6802 \global\csdef{eled@sectioning@#3#5}{\section[#1]{#2}}%
6803 \global\csdef{eled@sectmark@#3#5}{\let\edtext=\dummy@edtext}\
sectionmark{#1}}%Need for \pairs, because of using parbox.
6804 }%
6805 }%
6806 {\ifstrempy{#1}%
6807 {\global\csdef{eled@sectioning@#3#5}{\section*{#2}}}%
6808 {\global\csdef{eled@sectioning@#3#5}{\section*{#1}{#2}}}%Bug in
LaTeX!
6809 }
6810 \listcsgadd{eled@sections#5@@}{#3}%
6811 }
6812 \def\eled@subsection#1#2#3#4#5{%
6813 \ifstrempy{#4}%
6814 {\ifstrempy{#1}%
6815 {%
6816 \global\csdef{eled@sectioning@#3#5}{\subsection{#2}}%
6817 \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)
6818 }%
6819 {%
6820 \global\csdef{eled@sectioning@#3#5}{\subsection[#1]{#2}}%
6821 \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)
6822 }%
6823 }%
6824 {\ifstrempy{#1}%
6825 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#2}}}%
6826 {\global\csdef{eled@sectioning@#3#5}{\subsection*{#1}{#2}}}%Bug in
LaTeX!
6827 }
6828 \listcsgadd{eled@sections#5@@}{#3}%
6829 }
6830 \def\eled@subsubsection#1#2#3#4#5{%
6831 \ifstrempy{#4}%
6832 {\ifstrempy{#1}%
6833 {\global\csdef{eled@sectioning@#3#5}{\subsubsection{#2}}}%

```



```

6834     {\global\csdef{eled@sectioning@#3#5}{\subsubsection[#1]{#2}}}%
6835     }%
6836     {\ifstrempy{#1}%
6837     {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#2}}}%
6838     {\global\csdef{eled@sectioning@#3#5}{\subsubsection*{#1}{#2}}}%Bug
in LaTeX!
6839     }
6840     \listcsadd{eled@sections#5@@}{#3}%
6841     }
6842
6843 %

```

End of the conditional test about noeledsec option.

```

6844 }{}
6845 %

```

## 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.

```

6846 \def\normal@page@break{}
6847 %

```

**\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.

```

6848 \def\l@prev@pb{}
6849 \def\l@prev@nopb{}
6850 %

```

**\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.

```

6851 \newcommand{\ledpb}{\write\linenum@out{\string\led@pb}}
6852 \newcommand{\ledpbnum}[1]{\write\linenum@out{\string\led@pbnum{#1}}}
6853 \newcommand{\lednopb}{\write\linenum@out{\string\led@nopb}}

```

```

6854 \newcommand{\lednopbnum}[1]{\write\linenum@out{\string\led@nopbnum{#1}}}
6855 %

```

**\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.

```

6856 \newcommand{\led@pb}{\listxadd{\l@prev@pb}{\the\absline@num}}
6857 \newcommand{\led@pbnum}[1]{\listxadd{\l@prev@pb}{#1}}
6858 \newcommand{\led@nopb}{\listxadd{\l@prev@nopb}{\the\absline@num}}
6859 \newcommand{\led@nopbnum}[1]{\listxadd{\l@prev@nopb}{#1}}
6860 %

```

**\ledpbsetting** The \ledpbsetting macro only changes the value of \led@pb@macro, for which the default value is before.

**\led@pb@setting**

```

6861 \def\led@pb@setting{before}
6862 \newcommand{\ledpbsetting}[1]{\gdef\led@pb@setting{#1}}
6863 %

```

**\led@check@pb** The \led@check@pb and \led@check@nopb are called before or after each line. They check if a page break must occur, depending on the current line and on the content of \l@pb.

**\led@check@nopb**

```

6864 \newcommand{\led@check@pb}{\xifinlist{\the\absline@num}{\l@prev@pb}{\pagebreak[4]}}
6865 \newcommand{\led@check@nopb}{%
6866   \IfStrEq{\led@pb@setting}{before}{%
6867     \xifinlist{\the\absline@num}{\l@prev@nopb}{%
6868       {\numdef{\abs@prevline}{\the\absline@num-1}}%
6869       \xifinlist{\abs@prevline}{\normal@page@break}{%
6870         {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6871         {}}%
6872     }%
6873   }%
6874 }%
6875 \IfStrEq{\led@pb@setting}{after}{%
6876   \xifinlist{\the\absline@num}{\l@prev@nopb}{%
6877     \xifinlist{\the\absline@num}{\normal@page@break}{%
6878       {\nopagebreak[4]\enlargethispage{\baselineskip}}%
6879       {}}%
6880   }%
6881 }%
6882 }%
6883 }%
6884 }
6885 %

```

## 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.

```

6886 \newcommand{\check@pb@in@verse}{%
6887   \ifinstanza\iflednopbinverse\ifinserthangingsymbol% Using stanzas and
        enabling page breaks in verse control, while on a hanging verse.
6888   \ifnum\page@num=\last@page@num\else%If we have change page
6889   \IfStrEq{\led@pb@setting}{before}{%
6890     \numgdef{\abs@line@verse}{\the\absline@num-1}%
6891     \ledpbnum{\abs@line@verse}%
6892   }{}%
6893   \IfStrEq{\led@pb@setting}{after}{%
6894     \numgdef{\abs@line@verse}{\the\absline@num-1}%
6895     \lednopbnum{\abs@line@verse}%
6896   }{}%
6897   \fi%
6898   \fi\fi\fi%
6899 }
6900 %

```

## XXXIII Compatibility with eledmac

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

```

6901 \ifeledmaccompat%
6902
6903 \newcommand{\footnormalX}[1]{\arrangementX[#1]{normal}}%
6904 \newcommand{\footparagraphX}[1]{\arrangementX[#1]{paragraph}}%
6905 \newcommand{\foottwocolX}[1]{\arrangementX[#1]{twocol}}%
6906 \newcommand{\footthreecolX}[1]{\XarrangementX[#1]{threecol}}%
6907
6908 \unless\ifnocritical@
6909   \newcommand{\footnormal}[1]{\Xarrangement[#1]{normal}}%
6910   \newcommand{\footparagraph}[1]{\Xarrangement[#1]{paragraph}}%
6911   \newcommand{\foottwocol}[1]{\Xarrangement[#1]{twocol}}%
6912   \newcommand{\footthreecol}[1]{\Xarrangement[#1]{threecol}}%

```

```

6913 \let\hsizetwocol\Xhsizetwocol
6914 \let\hsizethreecol\Xhsizethreecol
6915 \let\bhookXnote\Xbhooknote
6916 \let\boxsymlinenum\Xboxsymlinenum
6917 \let\symlinenum\Xsymlinenum
6918 \let\beforenumberinfootnote\Xbeforenumber
6919 \let\afternumberinfootnote\Xafternumber
6920 \let\beforeXsymlinenum\XbeforeXsymlinenum
6921 \let\afterXsymlinenum\XafterXsymlinenum
6922 \let\inplaceofnumber\Xinplaceofnumber
6923 \let\Xlemmaseparator\lemmaseparator
6924 \let\afterlemmaseparator\Xafterlemmaseparator
6925 \let\beforelemmaseparator\Xbeforelemmaseparator
6926 \let\inplaceoflemmaseparator\Xinplaceoflemmaseparator
6927 \let\txbeforeXnotes\Xtxbeforenotes
6928 \let\afterXrule\Xafterrule
6929 \let\numberonlyfirstinline\Xnumberonlyfirstinline
6930 \let\numberonlyfirstintwoline\Xnumberonlyfirstintwoline
6931 \let\nonumberinfootnote\Xnonumberinfootnote
6932 \let\pstartinfootnote\Xpstart
6933 \let\pstartinfootnoteeverytime\Xpstarteverytime
6934 \let\onlyXpstart\Xonlypstart
6935 \let\Xnonumberinfootnote\Xnonumber
6936 \let\nonbreakableafternumber\Xnonbreakableafternumber
6937 \let\maxhXnotes\Xmaxhnotes
6938 \let\beforeXnotes\Xbeforenotes
6939 \let\boxlinenum\Xboxlinenum
6940 \let\boxlinenumalign\Xboxlinenumalign
6941 \let\boxstartlinenum\Xboxstartlinenum
6942 \let\boxendlinenum\Xboxendlinenum
6943 \let\twoline\Xtwoline
6944 \let\morethantwoline\Xmorethantwoline
6945 \let\twolinebutnotmore\Xtwolinebutnotmore
6946 \let\twolineonlyinsamepage\Xtwolineonlyinsamepage
6947 \fi
6948
6949 \unless\ifnofamiliar@
6950 \let\notesXwidthliketwocolumns\noteswidthliketwocolumnsX
6951 \fi
6952 \newcommandx{\parafootsep}[2][1,usedefault]{%
6953   \Xparafootsep[#1]{#2}%
6954   \parafootsepX[#1]{#2}
6955 }%
6956
6957 \newcommandx{\afternote}[2][1,usedefault]{%
6958   \Xafternote[#1]{#2}%
6959   \afternoteX[#1]{#2}%
6960 }%
6961
6962 \unless\ifnoend@

```

```

6963 \let\XendXtwolines\Xendtwolines
6964 \let\XendXmoreethantwolines\Xendmoreethantwolines
6965 \let\bhookXendnote\Xendbhooknote
6966 \let\boxXendlinenum\Xendboxlinenum%
6967 \let\boxXendlinenumalign\Xendboxlinenumalign%
6968 \let\boxXendstartlinenum\Xendboxstartlinenum%
6969 \let\boxXendendlinenum\Xendboxendlinenum%
6970 \let\XendXlemmaseparator\Xendlemmaseparator
6971 \let\XendXbeforelemmaseparator\Xendbeforelemmaseparator
6972 \let\XendXafterlemmaseparator\Xendafterlemmaseparator
6973 \let\XendXinplaceoflemmaseparator\Xendinplaceoflemmaseparator
6974 \fi
6975
6976 \AtBeginDocument{%
6977   \ifdef\lineref{}\let\lineref\edlineref}%
6978 }%
6979
6980
6981 \fi%
6982 %

```

</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>33</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>33</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. 102 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. 28) 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>34</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>34</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. 39). 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. 52.



## 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. 54 in order to know how to limit it.

### Appendix A.9.2 Multiple indices with memoir

Eledmac and ledmac used the specific indexing tools of the memoir class designed to produce multiple indices. However, eledmac could also use imakeidx or indextools tools independently of the memoir class. This system forced to maintain redundant code. Since reledmac, we use only the imakeidx or indextools tools.

Consequently: Users of memoir are invited to use indextool or imakeidx to produce multiple indices.

### Appendix A.9.3 Deprecated commands and options

The table of deprecated commands and their alternatives follows. Note that the way some commands must be used may have changed. Please read the handbook.

<i>Deprecated command</i>	<i>Replaced with</i>
<code>\addfootins</code>	<code>\newseries</code>
<code>\addfootinsX</code>	<code>\newseries</code>
<code>\critext</code>	<code>\edtext</code>
<code>\falseverse</code>	<code>\newverse</code>
<code>\interparanoteglue</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\ledchapter</code>	<code>\eledchapter</code>
<code>\ledsection</code>	<code>\eledsection</code>
<code>\ledsetnormalparstuff</code>	<code>\Xledsetnormalparstuff</code> and <code>\ledsetnormalparstuffX</code>
<code>\ledsubsection</code>	<code>\eledsubsection</code>
<code>\ledsubsubsection</code>	<code>\eledsubsubsection</code>
<code>\noeledsec</code>	Package option <code>noeledsec</code>
<code>\noendnotes</code>	Package option <code>noendnotes</code>
<code>\pageparbreak</code>	<code>\ledpb</code>

The `ledsecnolinenumber` option has been removed, because it was related to deprecated commands.

The `oldprintnpnumspace` option has been removed too, because it was related to a historical bug. The `\usingedtext` and `\usingcritext` commands are also deprecated.

#### Appendix A.9.4 `\renewcommand` replaced by `command`

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

<i>Deprecated <code>\renewcommand</code></i>	<i>Replaced with</i>
<code>\@led@extranofeet</code>	<code>\newseries</code>
<code>\apprefprefixmore</code>	<code>\setapprefprefixmore</code>
<code>\apprefprefixsingle</code>	<code>\setapprefprefixsingle</code>
<code>\endstanzaextra</code>	Optional argument of <code>\&amp;</code>
<code>\hangingsymbol</code>	<code>\sethangingsymbol</code>
<code>\ledfootinsdim</code>	<code>\Xmaxhnotes</code> and <code>\maxhnotesX</code>
<code>\parafootftmsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\notenumfont</code>	<code>\Xnotenumfont</code> , <code>\Xendnotenumfont</code> and <code>\notenumfontX</code>
<code>\notefontsetup</code>	<code>\Xnotefontsize</code> , <code>\Xendnotefontsize</code> and <code>\notefontsizeX</code>
<code>\sidenoteseq</code>	<code>\setsidenotsep</code>
<code>\startstanzahook</code>	Optional argument of <code>\stanza</code>
<code>\symplinenum</code>	<code>\Xsymplinenum</code>

#### Appendix A.9.5 Commands the names of which have been changed

In order to help the migration from eledmac to reledmac, you may load reledmac with `eledmac-compat` option. However, it is advised not to, and to change the command names themselves instead. In many cases, you use only a few of them, except the `\footparagraph` command.

<i>Old command</i>	<i>New command</i>
<code>\footparagraph</code>	<code>\Xarrangement</code>
<code>\footnormal</code>	<code>\Xarrangement</code>
<code>\foottwocol</code>	<code>\Xarrangement</code>
<code>\footthreecol</code>	<code>\Xarrangement</code>
<code>\footparagraphX</code>	<code>\arrangementX</code>
<code>\footnormalX</code>	<code>\arrangementX</code>
<code>\foottwocolX</code>	<code>\arrangementX</code>
<code>\footthreecolX</code>	<code>\arrangementX</code>
<code>\afterlemmaseparator</code>	<code>\Xafterlemmaseparator</code>
<code>\afternote</code>	<code>\Xafternote</code> and <code>\afternoteX</code>
<code>\afternumberinfootnote</code>	<code>\Xafternumber</code>
<code>\afterXrule</code>	<code>\Xafterrule</code>
<code>\afterXsymplinenum</code>	<code>\Xaftersymplinenum</code>
<code>\beforelemmaseparator</code>	<code>\Xbeforelemmaseparator</code>
<code>\beforenumberinfootnote</code>	<code>\Xbeforenumber</code>
<code>\beforeXnotes</code>	<code>\Xbeforenotes</code>
<code>\beforeXsymplinenum</code>	<code>\Xbeforesymplinenum</code>

<i>Old command</i>	<i>New command</i>
<code>\bhookXnote</code>	<code>\Xbhookendnote</code>
<code>\bhookXnote</code>	<code>\Xbhooknote</code>
<code>\boxendlinenum</code>	<code>\Xboxendlinenum</code>
<code>\boxlinenum</code>	<code>\Xboxlinenum</code>
<code>\boxlinenumalign</code>	<code>\Xboxlinenumalign</code>
<code>\boxstartlinenum</code>	<code>\Xboxstartlinenum</code>
<code>\boxsymlinenum</code>	<code>\Xboxsymlinenum</code>
<code>\boxXendlinenum</code>	<code>\Xendboxlinenum</code>
<code>\boxXendlinenumalign</code>	<code>\Xendboxlinenumalign</code>
<code>\boxXendstartlinenum</code>	<code>\boxXendstartlinenum</code>
<code>\letboxXendendlinenum</code>	<code>\Xendletboxendlinenum</code>
<code>\hsizetwocol</code>	<code>\Xhsizetwocol</code>
<code>\hsizethreecol</code>	<code>\Xhsizethreecol</code>
<code>\inplaceoflemmaseparator</code>	<code>\Xinplaceoflemmaseparator</code>
<code>\inplaceofnumber</code>	<code>\Xinplaceofnumber</code>
<code>\lemmaseparator</code>	<code>\Xlemmaseparator</code>
<code>\maxhXnotes</code>	<code>\Xmaxhnotes</code>
<code>\morethantwolines</code>	<code>\Xmorethantwolines</code>
<code>\nonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\notesXwidthliketwocolumns</code>	<code>\noteswidthliketwocolumnsX</code>
<code>\noXlemmaseparator</code>	<code>\Xnolemmaseparator</code>
<code>\numberonlyfirstinline</code>	<code>\Xnumberonlyfirstinline</code>
<code>\numberonlyfirstintwolines</code>	<code>\Xnumberonlyfirstintwolines</code>
<code>\nonbreakableafternumber</code>	<code>\Xnonbreakableafternumber</code>
<code>\onlyXpstart</code>	<code>\Xonlypstart</code>
<code>\parafootsep</code>	<code>\Xparafootsep</code> and <code>\parafootsepX</code>
<code>\pstartinfootnote</code>	<code>\Xpstart</code>
<code>\pstartinfootnoteeverytime</code>	<code>\Xpstarteverytime</code>
<code>\symlinenum</code>	<code>\Xsymlinenum</code>
<code>\twolines</code>	<code>\Xtwolines</code>
<code>\twolinesbutnotmore</code>	<code>\Xtwolinesbutnotmore</code>
<code>\twolinesonlyinsamepage</code>	<code>\Xtwolinesonlyinsamepage</code>
<code>\txtbeforeXnotes</code>	<code>\Xtxtbeforenotes</code>
<code>\XendXafterlemmaseparator</code>	<code>\Xendafterlemmaseparator</code>
<code>\XendXbeforelemmaseparator</code>	<code>\Xendbeforelemmaseparator</code>
<code>\XendXinplaceoflemmaseparator</code>	<code>\Xendinplaceoflemmaseparator</code>
<code>\XendXlemmaseparator</code>	<code>\Xendlemmaseparator</code>
<code>\XendXmorethantwolines</code>	<code>\Xendmorethantwolines</code>
<code>\XendXtwolines</code>	<code>\Xendtwolines</code>
<code>\Xnonumberinfootnote</code>	<code>\Xnonumber</code>
<code>\lineref</code>	<code>\edlineref</code>

**Appendix A.9.6 Endnotes**

With `reledmac`, there is now one auxiliary file for every endnotes set (`.Aend`, `.Bend`, `.Cend` etc.). If you have overridden `\doendnotes` (which you would not have done) you must adapt your code.

**Appendix A.9.7 Z Series**

The ‘Z’ series of notes has been removed. Only five series are provided now by default: A, B, C, D, E.

**Appendix A.9.8 Internal commands**

Users who have overridden internal commands, which is wrong, must adapt according to the following. Or better, they should not override any of such commands and use `reledmac` options instead.

- If you have modified `\Xfootfmt`, note that the fourth argument is now mandatory.
- `\unvxh` has been replaced with `\Xunvxh` and `\unvxhX` with two mandatory arguments.

**Appendix A.10 Migration to `reledmac` 2.1.0**

`Reledmac` 2.1.0 fix some bug 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 which use these commands to do it, which is not the original idea, must change to things:

1. Horizontal space is no more automatically added after the content of `\Xbhooknote`/`\bhooknoteX` argument. You must include it manually. So instead of `\Xbhooknote{content}`, use `\Xbhooknote{content }.`
2. Indent is no more automatically added before the content of `\Xbhooknote`/`\bhooknoteX` argument. If you want to keep it, add `\indent` in the argument of `\Xbhooknote`/`\bhooknoteX`.

## 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 `edstanzan.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>	39
<code>\@EDROWFILL@</code>	1
<code>\@adv</code>	1
<code>\@advancestanzanumber</code>	1
<code>\@apprefprefixmore</code>	1
<code>\@apprefprefixsingle</code>	1
<code>\@docclearpage</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	272
CLASSmemoir	166, 167, 210–212, 236, 272, 290, 340, 344
CLASSscrbook	344
COMMAND\*footnote	55

COMMAND\...\@footnotemark...	168
COMMAND\...d@ta	125
COMMAND\<hook	
@<series	201
COMMAND\<hookname	
<pseudoseries	203, 204
COMMAND\<type	
footfmt	158
COMMAND\@@line	151
COMMAND\@MM	139, 341
COMMAND\@Rlineflag	237, 341
COMMAND\@add@	261
COMMAND\@adv	89
COMMAND\@apprefprefixmore	220
COMMAND\@apprefprefixsingle	220
COMMAND\@bsphack	213
COMMAND\@docclearpage	211, 212, 334, 344
COMMAND\@doreinfeetX	344
COMMAND\@dprintingcolumns	341
COMMAND\@edindex@hyperref	237–239
COMMAND\@edtext@	105
COMMAND\@esphack	213
COMMAND\@fnpos	183
COMMAND\@footnotemark	167, 334, 344
COMMAND\@footnotetext	167, 168, 334
COMMAND\@gobble	104
COMMAND\@gobblefive	199, 342
COMMAND\@gobblefour	340
COMMAND\@gobblethree	333
COMMAND\@h	154
COMMAND\@hangingsymbol	240
COMMAND\@iiiminipage	228, 230, 333, 344
COMMAND\@iiiminpage	228
COMMAND\@l	339
COMMAND\@l@dttempcnta	127, 129, 135
COMMAND\@l@dttempcntb	129
COMMAND\@l@reg	339
COMMAND\@lab	86, 213, 215, 218, 333
COMMAND\@ldunboxmpfoot	230
COMMAND\@led@extranofeet	291
COMMAND\@ledinnote@command	234, 235
COMMAND\@lemma	108, 109
COMMAND\@lock	80, 240
COMMAND\@lopL	334
COMMAND\@lopR	334
COMMAND\@makecol	208, 210, 211, 344
COMMAND\@mpfnpos	183
COMMAND\@nl	86–90, 98, 215, 333, 334
COMMAND\@nl@reg	87, 288, 334, 339
COMMAND\@opXfeet	334



COMMAND\@opfeetX	344
COMMAND\@opxtrafeeti	344
COMMAND\@page	88, 215
COMMAND\@pend	334
COMMAND\@pendR	334
COMMAND\@ref	86, 95, 96, 99, 103
COMMAND\@ref@reg	95, 334
COMMAND\@reinserts	208–211, 344
COMMAND\@secondoftwo	56
COMMAND\@sect	275
COMMAND\@series	200
COMMAND\@set	90
COMMAND\@sidenotesep	227
COMMAND\@sw	96, 111, 114
COMMAND\@tag	105, 106, 109
COMMAND\@tempcnta	67
COMMAND\@tempcntb	67
COMMAND\@toksa	72
COMMAND\@toksb	72
COMMAND\@xloop	136
COMMAND\@xympar	222, 344
COMMAND\Aendnote	14, 22
COMMAND\Afootfmt	139
COMMAND\Afootgroup	139
COMMAND\Afootnote	7, 13, 21, 22, 24, 106, 146, 166, 185, 195, 343
COMMAND\Afootstart	139
COMMAND\AtEveryPend	16, 120, 340, 342, 343
COMMAND\AtEveryPstart	16, 340, 342, 343
COMMAND\Bendnote	14, 21
COMMAND\Bfootnote	7, 13, 166, 185, 195
COMMAND\Centering	35
COMMAND\Cfootnote	166
COMMAND\Columns	68, 143
COMMAND\Dfootnote	166
COMMAND\Efootnote	166
COMMAND\NR	275
COMMAND\Pages	68, 208, 210
COMMAND\ProcessOptionsX	60
COMMAND\RaggedLeft	35
COMMAND\RaggedRight	35
COMMAND\Stanza	339
COMMAND\Waklam	262
COMMAND\X@doreinfeet	209, 344
COMMAND\XXXXXXfmt	287
COMMAND\XXXXXfmt	287
COMMAND\Xafterlemmaseparator	33, 291
COMMAND\Xafternote	36, 290, 291
COMMAND\Xafternumber	31, 32, 291
COMMAND\Xafterrule	37, 185, 291, 339, 342
COMMAND\Xaftersymlinenum	32, 291

COMMAND\Xarrangement	29, 140, 141, 202, 291
COMMAND\Xarrangement@footparagraph	145
COMMAND\Xarrangement@normal	141
COMMAND\Xarrangement@paragraph	145
COMMAND\Xbeforelemmaseparator	33, 291
COMMAND\Xbeforenotes	36, 184, 291, 339, 342
COMMAND\Xbeforenumber	31, 32, 291
COMMAND\Xbeforesymlinenum	32, 291
COMMAND\Xbhookendnote	292
COMMAND\Xbhooknote	35, 292, 293, 344
COMMAND\Xboxendlinenum	32, 33, 292, 343
COMMAND\Xboxlinenum	32, 33, 292
COMMAND\Xboxlinenumalign	32, 33, 292, 343
COMMAND\Xboxstartlinenum	32, 33, 292, 343
COMMAND\Xboxsymlinenum	32, 292
COMMAND\Xcolalign	35, 342
COMMAND\Xdo@feet	208, 334, 344
COMMAND\Xend	199
COMMAND\XendXafterlemmaseparator	292
COMMAND\XendXbeforelemmaseparator	292
COMMAND\XendXinplaceoflemmaseparator	292
COMMAND\XendXlemmaseparator	292
COMMAND\XendXmorethantwolines	292
COMMAND\XendXtwolines	292
COMMAND\Xendafterlemmaseparator	33, 292
COMMAND\Xendafternote	37
COMMAND\Xendbeforelemmaseparator	33, 292
COMMAND\Xendbhooknote	35
COMMAND\Xendboxendlinenum	33, 343
COMMAND\Xendboxlinenum	33, 292, 341
COMMAND\Xendboxlinenumalign	33, 292, 343
COMMAND\Xendboxstartlinenum	33, 343
COMMAND\Xendinplaceoflemmaseparator	33, 292
COMMAND\Xendinplaceofnumber	32, 344
COMMAND\Xendinsertsep@	192
COMMAND\Xendlemmadisablefontselection	34
COMMAND\Xendlemmaseparator	23, 33, 292
COMMAND\Xendletboxendlinenum	292
COMMAND\Xendmorethantwolines	22, 30, 45, 292, 342
COMMAND\Xendmorethantwolinesappref	45
COMMAND\Xendnonumber	31, 344
COMMAND\Xendnote	188, 198, 199, 342
COMMAND\Xendnotefontsize	34, 291
COMMAND\Xendnotenumfont	34, 291
COMMAND\Xendparagraph	37, 339
COMMAND\Xendsep	37
COMMAND\Xendtwolines	22, 30, 45, 292, 342
COMMAND\Xendtwolinesappref	45
COMMAND\Xendtwolinesbutnotmore	30, 45, 342
COMMAND\Xendtwolinesbutnotmoreappref	45

COMMAND\Xendtwolinesonlyinsamepage	30, 45, 342
COMMAND\Xendtwolinesonlyinsamepageappref	45
COMMAND\Xfootfnt	293
COMMAND\Xfootgroup	144
COMMAND\Xfootins	143
COMMAND\Xfootnote	43, 105, 287, 336, 340–342
COMMAND\Xfootstarts	144
COMMAND\Xhangindent	34
COMMAND\Xsizethreecol	35, 292
COMMAND\Xsizetwocol	35, 203, 292
COMMAND\Xinplaceoflemmaseparator	33, 292
COMMAND\Xinplaceofnumber	32, 292, 341, 343
COMMAND\Xinsertparafootsep	149, 151
COMMAND\Xledsetnormalparstuff	289, 290, 342
COMMAND\Xlemmadisablefontselection	34
COMMAND\Xlemmaseparator	33, 205–207, 292
COMMAND\Xmaxhnotes	37, 291, 292, 339, 341
COMMAND\Xmorethantwolines	22, 30, 45, 207, 220, 292, 341
COMMAND\Xmorethantwolinesappref	45, 220
COMMAND\Xnolemmaseparator	33, 207, 292
COMMAND\Xnonbreakableafternumber	31, 292, 337
COMMAND\Xnonumber	31, 292
COMMAND\Xnonumberinfootnote	292
COMMAND\Xnotefontsize	34, 291
COMMAND\Xnotefontsize@(<s>)	149, 153, 154
COMMAND\Xnotenumberfont	34, 291
COMMAND\Xnoteswidthliketwocolumns	37, 340
COMMAND\Xnumberonlyfirstinline	29, 30, 83, 204–206, 292, 336, 341
COMMAND\Xnumberonlyfirstintwolines	30, 292, 336
COMMAND\Xonlypstart	31, 292, 336, 341
COMMAND\Xparafootsep	36, 83, 291, 292
COMMAND\Xparafootsep@series	149
COMMAND\Xparindent	34, 342
COMMAND\Xpstart	31, 289, 292, 336, 341
COMMAND\Xpstarteverytime	31, 289, 292, 341
COMMAND\Xragged	36
COMMAND\Xstanza	31, 41
COMMAND\Xstanzaseparator	31
COMMAND\Xsymlinenum	30, 36, 291, 292, 343
COMMAND\Xtwolines	22, 30, 45, 163, 164, 203, 207, 220, 292, 341
COMMAND\Xtwolinesappref	45, 203, 220
COMMAND\Xtwolinesbutnotmore	30, 45, 292, 342
COMMAND\Xtwolinesbutnotmoreappref	45, 204
COMMAND\Xtwolinesonlyinsamepage	30, 45, 292, 342
COMMAND\Xtwolinesonlyinsamepageappref	45
COMMAND\Xtxtbeforenotes	36, 292
COMMAND\Xunvxh	147, 293
COMMAND\&	291
COMMAND\absline@num	80, 126
COMMAND\accent	104

COMMAND\actionlines@list	81, 127
COMMAND\actions@list	81
COMMAND\add@inserts	81, 134
COMMAND\add@inserts@next	134, 135
COMMAND\add@penalties	126, 135
COMMAND\addcontentsline	269
COMMAND\addfootins	287, 290
COMMAND\addfootinsX	287, 290
COMMAND\advancelabel@refs	214
COMMAND\advanceline	19, 20, 82, 89, 100, 344
COMMAND\advancepageno	208
COMMAND\affixlin@num	227
COMMAND\affixline@num	128, 131, 132, 334
COMMAND\affixpstart@num	132
COMMAND\afterXrule	291
COMMAND\afterXsymlinenum	291
COMMAND\aftergroup	103, 107
COMMAND\afterlemmaseparator	291
COMMAND\afternote	291
COMMAND\afternoteX	36, 290, 291
COMMAND\afternumberinfootnote	291
COMMAND\afterruleX	37, 339, 342
COMMAND\applabel	44, 216, 217, 220, 342
COMMAND\appref	45, 220, 221
COMMAND\apprefprefixmore	291
COMMAND\apprefprefixsingle	291
COMMAND\apprefwithpage	45, 220, 221
COMMAND\arrangementX	29, 169, 202, 291
COMMAND\arrangementX@normal	173
COMMAND\at@every@pend	120
COMMAND\autopar	15, 16, 117, 120, 121, 181, 335, 337, 338, 342
COMMAND\ballast	55
COMMAND\ballast@count	126, 135
COMMAND\baselineskip	29, 145, 146, 149
COMMAND\beforeXnotes	291
COMMAND\beforeXsymlinenum	291
COMMAND\beforeelectedchapter	8, 52, 270
COMMAND\beforelemmaseparator	291
COMMAND\beforenotesX	36, 338, 339, 342
COMMAND\beforenumberinfootnote	291
COMMAND\begin	247
COMMAND\beginnumbering	14, 16, 17, 68, 69, 71, 79, 84, 98, 121, 187, 336, 339, 343, 344
COMMAND\bf	336
COMMAND\bfseries	34, 336
COMMAND\bhookXnote	292
COMMAND\bhooknoteX	35, 293, 344
COMMAND\body	241
COMMAND\bodyfootmarkA	27
COMMAND\boxXendlinenum	292
COMMAND\boxXendlinenumalign	292

COMMAND\boxXendstartlinenum	292
COMMAND\boxendlinenum	292
COMMAND\boxlinefootnote	160
COMMAND\boxlinenum	292
COMMAND\boxlinenumalign	292
COMMAND\boxstartlinenum	292
COMMAND\boxsymlinenum	292
COMMAND\break	147
COMMAND\brokenpenalty	135
COMMAND\centering	35
COMMAND\ch@ck@l@ck	335
COMMAND\ch@cksub@l@ck	131, 335
COMMAND\chapter	51, 272, 339, 342, 344
COMMAND\chaptermark	269
COMMAND\check@pb@in@verse	283
COMMAND\colalignX	35, 342
COMMAND\collect@body	248
COMMAND\colorbox	56
COMMAND\columns	37
COMMAND\columnwidth	146, 340
COMMAND\command names	203, 204
COMMAND\copyright	104
COMMAND\correct@Xfootins@box	341
COMMAND\correct@footinsX@box	341
COMMAND\count	152, 153
COMMAND\critex	335
COMMAND\critext	110, 286, 287, 290
COMMAND\csname	60, 113
COMMAND\ctab	263, 268
COMMAND\ctabtext	268
COMMAND\dc col	257
COMMAND\def	58
COMMAND\detokenize	113
COMMAND\dimen	152
COMMAND\discretionary	147
COMMAND\displaywidowpenalty	135
COMMAND\do@actions	126, 128, 335
COMMAND\do@actions@fixedcode	334
COMMAND\do@actions@next	126, 127
COMMAND\do@ballast	126, 135
COMMAND\do@feetX	344
COMMAND\do@insidelinehook	337
COMMAND\do@line	81, 103, 119, 122, 124, 125, 134, 135, 240, 335, 337, 339
COMMAND\do@linehook	335
COMMAND\do@lockoff	82
COMMAND\do@lockon	82
COMMAND\dodoreinextrafeet	333
COMMAND\doendnotes	22, 192, 293, 342
COMMAND\doendnotesbysection	23, 192, 199, 343
COMMAND\doinsidelinehook	20, 340

COMMAND\dolinehook	20, 340
COMMAND\doreintrafeeti	344
COMMAND\doreintrafeetii	344
COMMAND\doxtrafeet	208, 333
COMMAND\doxtrafeeti	344
COMMAND\doxtrafeetii	344
COMMAND\dummy@ref	103
COMMAND\edaftertab	51, 263, 264
COMMAND\edatleft	50, 261
COMMAND\edatright	50, 51, 261
COMMAND\edbeforetab	51, 263
COMMAND\edfilldimen	261
COMMAND\edfont@info	109
COMMAND\edindex	48, 233, 237–239, 251, 337, 340, 341, 344
COMMAND\edindexlab	48
COMMAND\edlabel	43, 44, 104, 213–215, 217, 221, 233, 251, 333, 336–338, 341
COMMAND\edlineref	43, 213, 292, 341, 343
COMMAND\edmakelabel	44, 221
COMMAND\edpageref	43, 213, 217, 221
COMMAND\edrowfill	262
COMMAND\edtabcolsep	256
COMMAND\edtext	6, 21–27, 38, 43, 44, 49, 55, 56, 81, 95, 96, 99, 102–110, 112, 114, 115, 216, 217, 219, 251, 253, 272, 286, 287, 290, 334, 335, 337, 339–343
COMMAND\edtext@level	343
COMMAND\edvertdots	51, 261
COMMAND\edvertline	51, 261
COMMAND\elechapter	52
COMMAND\eled@sectioning@out	276
COMMAND\eledchapter	52, 290, 340, 344
COMMAND\eledchapter*	52
COMMAND\eledmac@error	333
COMMAND\eledsection	6, 13, 52, 103, 124, 270, 290, 341
COMMAND\eledsection*	52
COMMAND\eledsubsection	52, 290
COMMAND\eledsubsection*	52
COMMAND\eledsubsubsection	52, 290
COMMAND\eledsubsubsection*	52
COMMAND\eledxxx	8, 53, 271, 276, 339
COMMAND\eledxxxx	270
COMMAND\else	232, 270
COMMAND\empty	67, 130, 213
COMMAND\end	247
COMMAND\end@lemmas	103
COMMAND\endashchar	38, 157
COMMAND\endgraf	119, 149, 181
COMMAND\endlock	19, 82, 101, 245
COMMAND\endminipage	228, 230, 333, 344
COMMAND\endnotes	342
COMMAND\endnumbering	14, 17, 68, 70, 71, 335, 343
COMMAND\endprint	187, 189, 199, 289

COMMAND\endstanzaextra	291
COMMAND\endsub	19, 82, 100
COMMAND\everypar	121
COMMAND\extensionchars	54, 68
COMMAND\f@x@l@cks	335
COMMAND\falseverse	290, 337, 339
COMMAND\fi	270
COMMAND\firstlinenum	18, 129, 335
COMMAND\firstsublinenum	18, 335
COMMAND\fix@page	87, 88, 334
COMMAND\flag@end	99, 108, 339
COMMAND\flag@start	99, 108, 339, 340
COMMAND\flagstanza	42
COMMAND\floatingpenalty	139, 341
COMMAND\flush@notes	136
COMMAND\fnpos	183, 338
COMMAND\footfmt	139, 141
COMMAND\footfmt...	169
COMMAND\footfootmarkA	27
COMMAND\footfudgefactor	147
COMMAND\footfudgefiddle	55, 145, 146, 333
COMMAND\footgroup	139
COMMAND\footins	143
COMMAND\footnormal	203, 291, 334
COMMAND\footnormalX	291
COMMAND\footnote	27, 55, 166, 167, 288, 334
COMMAND\footnote@lang	157
COMMAND\footnoteA	14, 27
COMMAND\footnoteB	14
COMMAND\footnoteC	21
COMMAND\footnoteE	27
COMMAND\footnoteX	7, 197
COMMAND\footnoteXmk	207
COMMAND\footnotelang@lua	138
COMMAND\footnotelang@poly	138
COMMAND\footnoteoption@	137
COMMAND\footnoterule	152
COMMAND\footnotesize	34
COMMAND\footparagraph	145, 203, 291, 339
COMMAND\footparagraphX	178, 291, 339
COMMAND\footsplitskips	335, 341
COMMAND\footstart	139, 143, 152
COMMAND\footstrut	149
COMMAND\footthreecol	291
COMMAND\footthreecolX	291, 342
COMMAND\foottwocol	291
COMMAND\foottwocolX	291, 342
COMMAND\fulllines@	207
COMMAND\fullstop	38
COMMAND\get@edindex@hyperref	237

COMMAND\get@edindex@ledinnote@command	234
COMMAND\get@index@command	338
COMMAND\get@linelistfile	335
COMMAND\getline@num	125, 127
COMMAND\gl@p	73
COMMAND\global	86
COMMAND\globaldefs	86
COMMAND\hangindentX	34, 342
COMMAND\hangingsymbol	291, 335
COMMAND\hbox	147
COMMAND\hfill	338
COMMAND\hidenumering	20, 94, 342
COMMAND\hline	49
COMMAND\hrulefill	262
COMMAND\hsize	29, 143, 145–147, 153, 155, 182, 334, 340
COMMAND\hsizethreecol	292
COMMAND\hsizethreecolX	35
COMMAND\hsizetwocol	35, 292
COMMAND\hyperlinkR	237
COMMAND\hyperlinkformat	237
COMMAND\hyperlinkformatR	237
COMMAND\if@RTL	61
COMMAND\if@edtext@	340, 343
COMMAND\if@eled@sectioning	271
COMMAND\if@noneed@Footnote	99
COMMAND\ifbypage@	73
COMMAND\ifbypage@R	73
COMMAND\ifbypstart@	73
COMMAND\ifbypstart@R	73
COMMAND\iffirst@linenum@out@	97, 98
COMMAND\ifinserthangingsymbol	240
COMMAND\ifinstanza	240
COMMAND\ifistwofollowinglines	164
COMMAND\ifl@d@Xmorethantwolines	161, 342
COMMAND\ifl@d@Xtwolines	161
COMMAND\ifl@d@dash	161
COMMAND\ifl@d@elin	161
COMMAND\ifl@d@esl	161
COMMAND\ifl@d@pnum	161
COMMAND\ifl@d@ssub	161
COMMAND\ifl@dend@X	198
COMMAND\ifl@dmemoir	333
COMMAND\ifl@dpaging	340
COMMAND\ifl@dpairing	67, 335
COMMAND\ifl@dprintingpages	341
COMMAND\ifl@dskipnumber	129
COMMAND\ifl@dstartendok	262
COMMAND\ifl@imakeidx	61
COMMAND\ifledRcol	68, 336
COMMAND\ifledRcol@	68, 339



COMMAND\iflemmacommand@	341
COMMAND\ifnoledgroup@	232
COMMAND\ifnoteschanged@	83
COMMAND\ifnumberedpar@	117
COMMAND\ifnumbering	68, 71
COMMAND\ifnumberingR	68, 336
COMMAND\ifnumberline	108, 129
COMMAND\ifpst@rted	335
COMMAND\ifpst@rtedL	69
COMMAND\ifseriesbefore	201
COMMAND\ifsublines@	80, 92
COMMAND\iftrue	343
COMMAND\ifvmode	214
COMMAND\ifxxx	270
COMMAND\ignorespaces	106, 107
COMMAND\imki@wrindexentry	61
COMMAND\immediate	97, 98, 187
COMMAND\indent	16, 121, 293
COMMAND\indtl@wrindexentry	61
COMMAND\initnumbering@quote	268, 344
COMMAND\initnumbering@reg	335
COMMAND\initnumbering@sectcmd	344
COMMAND\inplaceoflemmaseparator	292
COMMAND\inplaceofnumber	292
COMMAND\insert	134, 139, 141, 169
COMMAND\insert@count	95, 99, 106
COMMAND\insert@countR	106
COMMAND\insertthangingsymbol	338
COMMAND\insertlines@list	81, 95
COMMAND\insertparafootsepX	181
COMMAND\inserts@list	103, 117, 134, 146
COMMAND\interAfootnotelinepenalty	334
COMMAND\interfootnotelinepenalty	334
COMMAND\interlinepenalty	139
COMMAND\interparanoteglue	290
COMMAND\justifying	35
COMMAND\l@advance@parledegroupp@beforenormalnotes	344
COMMAND\l@d@@wrindexhyp	340
COMMAND\l@d@add	110
COMMAND\l@d@end	187, 198
COMMAND\l@d@nums	106, 108, 110, 161
COMMAND\l@d@section	187
COMMAND\l@d@set	90, 101
COMMAND\l@dampcount	253
COMMAND\l@dbfnote	168, 334
COMMAND\l@dcheckstartend	262
COMMAND\l@dchset@num	90
COMMAND\l@dcolcount	253, 254
COMMAND\l@dcollect@@body	247
COMMAND\l@dcollect@body	247

COMMAND\l@dcsnote	339
COMMAND\l@dcsnotetext	125, 225
COMMAND\l@dcsnotetext@l	125, 225, 226
COMMAND\l@dcsnotetext@r	125, 225
COMMAND\l@ddodorextrafeet	209, 333
COMMAND\l@ddoxtrafeet	209, 333
COMMAND\l@emptyd@ta	335
COMMAND\l@dend@close	187
COMMAND\l@dend@open	187
COMMAND\l@dend@stuff	187
COMMAND\l@denbody	247
COMMAND\l@dfeetbeginmini	334
COMMAND\l@dfeetendmini	334
COMMAND\l@dgetline@margin	335
COMMAND\l@dgetlock@disp	335
COMMAND\l@dgetref@num	218, 219
COMMAND\l@dgetsidenote@margin	222, 335
COMMAND\l@dgobbeloptarg	340
COMMAND\l@dgonblearg	340
COMMAND\l@dgonbleoptarg	252
COMMAND\l@dlabel@parse	218, 219
COMMAND\l@dld@ta	128, 131
COMMAND\l@dlp@rbox	226
COMMAND\l@dlsn@te	335
COMMAND\l@dlsnote	339
COMMAND\l@dmake@labels	215
COMMAND\l@dnumpstartsL	69, 335
COMMAND\l@dp@rsefootspec	161
COMMAND\l@dparsefootspec	161
COMMAND\l@dpush@begins	247
COMMAND\l@drd@ta	128, 131
COMMAND\l@dref@undefined	218
COMMAND\l@drsn@te	335
COMMAND\l@drsnote	339
COMMAND\l@dtabaddcols	261
COMMAND\l@dtabnoexpands	333
COMMAND\l@dumboxmpfoot	344
COMMAND\l@dunboxmpfoot	335
COMMAND\l@dzeropenalties	335, 340
COMMAND\l@pb	282
COMMAND\l@prev@nopb	281
COMMAND\l@prev@pb	281
COMMAND\l@reg	288
COMMAND\label	16, 44, 48, 213, 214, 219
COMMAND\label@refs	213
COMMAND\labelstarttrue	16, 336
COMMAND\labelref@list	213, 215
COMMAND\language	147
COMMAND\last@page@num	334
COMMAND\lastbox	121

COMMAND\lastskip	100
COMMAND\leavevmode	16, 121
COMMAND\led@check@nopb	282
COMMAND\led@check@pb	282
COMMAND\led@nopb	281–283
COMMAND\led@nopbnum	281, 282
COMMAND\led@pb	281–283
COMMAND\led@pb@macro	282
COMMAND\led@pbnum	281, 282
COMMAND\ledRflag	237
COMMAND\ledchapter	290, 337
COMMAND\ledfootinsdim	291
COMMAND\ledinnernote	45, 224, 339
COMMAND\ledinnote	235, 289, 343
COMMAND\ledinnotemark	43, 289, 342
COMMAND\ledleftnote	46, 224
COMMAND\ledlinenum	78, 335
COMMAND\ledllfill	125
COMMAND\ledsnotesep	46
COMMAND\ledsnotewidth	46
COMMAND\lednopb	53, 281
COMMAND\lednopbinverse	283
COMMAND\lednopbinversetrue	41, 53
COMMAND\lednopbnum	281
COMMAND\ledouternote	45, 224, 339
COMMAND\ledpb	53, 281, 290
COMMAND\ledpbnum	281
COMMAND\ledpbsetting	54, 282, 344
COMMAND\ledrightnote	46, 224
COMMAND\ledrsnotesep	46
COMMAND\ledrsnotewidth	46
COMMAND\ledsection	290
COMMAND\ledsectnomark	269
COMMAND\ledsectnotoc	269
COMMAND\ledsetnormalparstuff	289, 290, 342
COMMAND\ledsetnormalparstuff@common	182
COMMAND\ledsetnormalparstuffX	289, 290, 342
COMMAND\ledsidenote	45, 46, 224–226
COMMAND\ledsubsection	290
COMMAND\ledsubsubsection	290
COMMAND\ledxxx	339
COMMAND\left	50
COMMAND\leftctab	263
COMMAND\leftheadline	78
COMMAND\leftlinenum	19, 78, 333, 335
COMMAND\leftltab	263
COMMAND\leftnoteupfalse	46
COMMAND\leftpstartnum	133
COMMAND\leftftab	263
COMMAND\leftsidenote	225

COMMAND\leftskip	143, 146, 147
COMMAND\lemma	2, 22–27, 102, 105–107, 109, 110, 112, 286, 335, 336, 343, 344
COMMAND\lemmaseparator	292
COMMAND\let	24, 38, 245, 333
COMMAND\letboxXendendlinenum	292
COMMAND\line	151, 154
COMMAND\line@list	81, 96, 108
COMMAND\line@list@stuff	69, 84, 98, 333, 335
COMMAND\line@list@version	86
COMMAND\line@margin	75, 130, 222
COMMAND\line@num	79, 80, 82, 129, 333
COMMAND\line@set	110
COMMAND\lineation	18, 74
COMMAND\linenum	22–24, 43–45, 102, 110, 217, 219, 222, 286
COMMAND\linenum@out	97, 213, 215
COMMAND\linenumberlist	18, 67, 130, 333
COMMAND\linenumberstyle	20, 78, 333
COMMAND\linenumincrement	18, 335
COMMAND\linenummargin	18, 75, 222
COMMAND\linenumr@p	78, 333, 335
COMMAND\linenumrep	78, 335
COMMAND\linenumsep	19, 46, 78, 223
COMMAND\lineref	213, 217, 221, 292, 341
COMMAND\list@clear	72
COMMAND\list@clearing@reg	335
COMMAND\list@create	72
COMMAND\lock@disp	77
COMMAND\lock@off	93
COMMAND\lock@on	92
COMMAND\lockdisp	19, 77
COMMAND\loop	136, 241
COMMAND\ltab	263, 264, 268
COMMAND\ltabtext	268
COMMAND\m@mmf@prepare	167
COMMAND\makeatletter	125
COMMAND\makehboxofhboxes	148, 149
COMMAND\makeindex	47, 237
COMMAND\makelabel	221
COMMAND\managestanza@modulo	242
COMMAND\marginpar	45, 55, 222, 334
COMMAND\marginparwidth	46, 223
COMMAND\markboth	124
COMMAND\mathchardef	241
COMMAND\maxhXnotes	292
COMMAND\maxhnotesX	37, 291, 338, 339, 341, 342
COMMAND\maxlinesinpar@list	84
COMMAND\measurebody	264
COMMAND\measuretbody	265
COMMAND\memorybreak	17
COMMAND\morenoexpands	56, 102, 104

COMMAND\morethantwolines	292
COMMAND\mpfnpos	183, 338
COMMAND\mpnormalfootgroup	334
COMMAND\mpnormalvfootnote	334
COMMAND\multfootsep	27, 166
COMMAND\multiplefootnotemark	166
COMMAND\musixtex	339
COMMAND\n@num	335, 342
COMMAND\n@num@ref	342
COMMAND\new@line	98, 334
COMMAND\newcommand	24, 58, 166, 215
COMMAND\newcommandx	24
COMMAND\newhookcommand@series	203, 204, 342
COMMAND\newhookcommand@series@reload	204
COMMAND\newhookcommand@toggle@reload	204, 340
COMMAND\newhooktoggle@series	204, 342
COMMAND\newif	342
COMMAND\newlinechar	198
COMMAND\newseries	28, 287, 290, 291
COMMAND\newseries@	193, 202
COMMAND\newverse	41, 290, 339
COMMAND\next	241
COMMAND\next@action	84
COMMAND\next@actionline	84
COMMAND\next@insert	134
COMMAND\nl@regR	87
COMMAND\no@expands	56, 109, 333
COMMAND\noXlemmaseparator	292
COMMAND\nobreak	160
COMMAND\nocritical	194
COMMAND\noeledsec	53, 290
COMMAND\noendnotes	290
COMMAND\noexpand	288
COMMAND\nofamiliar	206
COMMAND\noindent	16, 121
COMMAND\nomk@	207
COMMAND\nonbreakableafternumber	292
COMMAND\nonum@	207
COMMAND\nonumberinfootnote	292
COMMAND\normal@footnotemarkX	169
COMMAND\normal@page@break	281
COMMAND\normal@pars	181
COMMAND\normalbfnoteX	335
COMMAND\normalbodyfootmarkX	169
COMMAND\normalfootfmt	38, 142, 149, 157, 187
COMMAND\normalfootfmtX	170
COMMAND\normalfootfootmarkX	170
COMMAND\normalfootgroup	144
COMMAND\normalfootgroupX	171
COMMAND\normalfootnoterule	140

COMMAND\normalfootstart	143, 146
COMMAND\normalfootstartX	171
COMMAND\normalvfootnote	141
COMMAND\normalvfootnoteX	169
COMMAND\notbool	270
COMMAND\notfontsetup	291
COMMAND\notfontsizeX	34, 291
COMMAND\notenumfont	291
COMMAND\notenumfontX	34, 291
COMMAND\notesXwidthliketwocolumns	292
COMMAND\noteswidthliketwocolumnsX	37, 292, 340, 342
COMMAND\num@lines	117, 135
COMMAND\numberlinefalse	17
COMMAND\numberlinetrue	17
COMMAND\numberonlyfirstinline	201, 292
COMMAND\numberonlyfirstintwolines	292
COMMAND\numberpstartfalse	16
COMMAND\numberpstarttrue	16, 31, 289, 335, 344
COMMAND\numberstanza	31
COMMAND\numberstanzafalse	41
COMMAND\numberstanzatrue	41
COMMAND\numlabfont	19, 38, 78, 79
COMMAND\one@line	117
COMMAND\onlyXpstart	292
COMMAND\page@action	82, 91
COMMAND\page@start	82, 335
COMMAND\pagecontents	82
COMMAND\pagelinesep	47
COMMAND\pageno	208
COMMAND\pageparbreak	290
COMMAND\pageref	44, 217
COMMAND\par	121, 181
COMMAND\par@line	117, 135
COMMAND\para@footgroup	146
COMMAND\para@footgroupX	180
COMMAND\para@footsetup	145, 333
COMMAND\para@footsetupX	178, 333, 340
COMMAND\para@vfootnote	149
COMMAND\para@vfootnoteX	179
COMMAND\parafootfmt	148, 149
COMMAND\parafootfmtX	180
COMMAND\parafootftm	151
COMMAND\parafootftmX	181
COMMAND\parafootftmsep	291
COMMAND\parafootsep	36, 292, 338, 343
COMMAND\parafootsepX	36, 83, 291, 292
COMMAND\parafootstart	146
COMMAND\parafootstartX	179
COMMAND\paravfootnote	146
COMMAND\parfillskip	149

COMMAND\parindentX	34
COMMAND\parshape	55
COMMAND\parskip	121
COMMAND\pausenumbering	17, 71, 84, 86, 122, 338, 340
COMMAND\penalty	149
COMMAND\pend	2, 6, 15–18, 20, 52, 53, 100, 103, 105, 111, 117, 119–122, 132–134, 288, 338, 339
COMMAND\preXnotes	36, 185, 342
COMMAND\preXnotes@	143, 185, 336
COMMAND\prenotesX	36, 186, 342
COMMAND\prepare@preXnotes	184
COMMAND\prev@nopb	282
COMMAND\prev@pb	282
COMMAND\prevlineX	83
COMMAND\prevpageX@num	83
COMMAND\print@Xfootnoterule	342
COMMAND\print@Xnotes	208, 210
COMMAND\print@Xnotes@forpages	341
COMMAND\print@eledsection	124
COMMAND\print@footnoteXrule	342
COMMAND\print@leftmargin@eledsection	271
COMMAND\print@line	123
COMMAND\print@notesX@forpages	341
COMMAND\print@rightmargin@eledsection	271
COMMAND\printendlines	189, 221, 333, 335
COMMAND\printlinefootnote	158, 159, 341
COMMAND\printlinefootnotearea	160, 341
COMMAND\printlinefootnotenumbers	158
COMMAND\printlines	142, 157, 161, 162, 189, 221, 333, 335, 342
COMMAND\printnpnum	23, 289
COMMAND\printpstart	158
COMMAND\protect	104, 288
COMMAND\providecommand	166, 333
COMMAND\pstart	2, 6, 15–18, 20, 51–53, 90, 100, 101, 105, 111, 117, 119–121, 124, 134, 288, 335, 336, 338–340, 342–344
COMMAND\pstartinfootnote	292
COMMAND\pstartinfootnoteeverytime	292
COMMAND\pstartnum	133
COMMAND\pstartref	43, 213, 218, 338
COMMAND\pstarts	336
COMMAND\raggedX	36
COMMAND\raggedleft	35
COMMAND\raggedright	35
COMMAND\raw@text	117
COMMAND\rbracket	33, 38
COMMAND\read@linelist	84, 85
COMMAND\ref	44, 48
COMMAND\relax	16, 90, 126, 134, 245, 252, 288
COMMAND\renewcommand	55, 291
COMMAND\resetprevline@	83
COMMAND\resetprevpage@	83

COMMAND\resumenumbering	17, 68, 71, 84, 86, 122, 335, 339, 340
COMMAND\right	50
COMMAND\rightctab	263
COMMAND\rightlinenum	19, 78, 333, 335
COMMAND\rightltab	264
COMMAND\rightnoteupfalse	46
COMMAND\rightrtab	264
COMMAND\rightsidenote	225
COMMAND\rightskip	143, 146, 147, 149
COMMAND\rightstartnum	133
COMMAND\rigidbalance	151, 152, 154
COMMAND\robustify	30
COMMAND\rtab	263, 264, 268
COMMAND\rtabtext	265, 268
COMMAND\sameword	25–27, 111–113, 115, 341, 343
COMMAND\sameword@inedtext	112
COMMAND\saweword	112
COMMAND\scriptsize	79
COMMAND\section	51, 335
COMMAND\section@num	68
COMMAND\sectionmark	269
COMMAND\select@lemmafnt	38, 137
COMMAND\series	193
COMMAND\series@	193
COMMAND\seriesatbegin	28, 200, 342
COMMAND\seriesatend	28, 200, 343
COMMAND\set@line	108
COMMAND\set@line@action	82, 91
COMMAND\setapprefprefixmore	45, 291
COMMAND\setapprefprefixsingle	45, 291
COMMAND\setcommand@series	202
COMMAND\sethangingsymbol	40, 240, 291
COMMAND\sethanginsymbol	39
COMMAND\setistwofollowinglines	164
COMMAND\setl@dlprbox	226
COMMAND\setline	19, 20, 82, 86, 90, 100, 104, 119, 344
COMMAND\setlinenum	20, 86, 90, 101, 333
COMMAND\setprintendlines	189, 191, 335
COMMAND\setprintlines	162, 164, 189, 335
COMMAND\setsidenotessep	46
COMMAND\setsidenotsep	291
COMMAND\setstanzaindent	242
COMMAND\setstanzaindents	40, 241, 288
COMMAND\setstanzapenalties	241
COMMAND\setstanzavalues	241
COMMAND\settoggle@series	202, 336, 340
COMMAND\showlemma	103, 334
COMMAND\showwordrank	27, 113
COMMAND\sidenote@margin	334
COMMAND\sidenotemargin	45, 334, 339



COMMAND\sidenotesep	291
COMMAND\sidepstartnumtrue	16
COMMAND\skip	143
COMMAND\skipnumbering	20, 94, 101, 335, 343
COMMAND\skipnumbering@reg	343
COMMAND\small	34
COMMAND\special	11
COMMAND\splitmaxdepth	139, 153
COMMAND\splitoff	151
COMMAND\splittopskip	139, 153, 154
COMMAND\stanza	19, 20, 41, 244, 291
COMMAND\stanza@hang	244
COMMAND\stanza@line	244
COMMAND\stanzaindent	40, 242, 341
COMMAND\stanzaindent*	40
COMMAND\stanzaindentbase	241
COMMAND\stanzanumwrapper	41
COMMAND\start	53
COMMAND\startlock	19, 82, 101, 245
COMMAND\startstanzahook	291
COMMAND\startsub	19, 82, 100
COMMAND\strip@pt	146
COMMAND\strutbox	153
COMMAND\sub@action	82, 92
COMMAND\sub@lock	80
COMMAND\sub@off	89, 215
COMMAND\sub@on	89, 215
COMMAND\subline@num	79, 80, 82
COMMAND\sublinenum@rep	333
COMMAND\sublinenumberstyle	20, 78, 333
COMMAND\sublinenumincrement	18
COMMAND\sublinenumr@p	78, 333, 335
COMMAND\sublinenumrep	78, 335
COMMAND\sublineref	43, 213, 218
COMMAND\subsectionmark	269
COMMAND\sw@inthisedtext	105
COMMAND\sw@list@inedtext	109, 115
COMMAND\symlinenum	292
COMMAND\symplinenum	291
COMMAND\sza@penalty	244
COMMAND>tag	341
COMMAND\text	286
COMMAND\textcolor	56
COMMAND\the	333
COMMAND\thefootnoteA	27
COMMAND\thefootnoteX	337
COMMAND\thelabidx	238, 239
COMMAND\thepage	87
COMMAND\thepstart	16
COMMAND\thepstartL	336

COMMAND\thepstartR	336
COMMAND\thestanza	41
COMMAND\this@line@list@version	97
COMMAND\threecolfootfmt	153
COMMAND\threecolfootfmtX	177
COMMAND\threecolfootgroup	152
COMMAND\threecolfootgroupX	177
COMMAND\threecolfootsetup	152
COMMAND\threecolfootsetupX	176
COMMAND\threecolvfootnote	153
COMMAND\threecolvfootnoteX	176
COMMAND\twocolfootfmtX	175
COMMAND\twocolfootgroupX	175
COMMAND\twocolfootsetupX	174
COMMAND\twocolvfootnoteX	174
COMMAND\twolines	201, 292
COMMAND\twolines@A	201
COMMAND\twolines@B	201
COMMAND\twolines@C	201
COMMAND\twolinesbutnotmore	292
COMMAND\twolinesonlyinsamepage	292
COMMAND\txbeforeXnotes	292
COMMAND\unhbox	147
COMMAND\unpenalty	148, 149
COMMAND\unskip	149
COMMAND\unvxh	148, 149, 293
COMMAND\unvxhX	293
COMMAND\upbracefill	262
COMMAND\usingcritext	287, 290
COMMAND\usingdtext	287, 290
COMMAND\VAfootnote	139
COMMAND\variant	24
COMMAND\ vbox	119, 121, 147, 151, 184
COMMAND\ vfootnote	139, 143, 146, 153
COMMAND\ vl@dbfnote	168, 334
COMMAND\ vnumfootnoteX	335
COMMAND\ vsplit	135
COMMAND\ waklam	262
COMMAND\ waklamec	262
COMMAND\ wapunktel	262
COMMAND\ wastricht	262
COMMAND\ wrap@edcrossref	217, 340
COMMAND\ x...	44
COMMAND\ xdef	72, 245
COMMAND\ xleft@appenditem	73, 103
COMMAND\ xlineref	43
COMMAND\ xpageref	43
COMMAND\ xpstartref	43, 338
COMMAND\ xright@appenditem	72, 73
COMMAND\ xsublineref	43

COMMAND\xxref	44, 219, 222, 338, 341, 342
COMMAND\zz@@@	333
ENVIRONMENTedarrayc	268
ENVIRONMENTedarrayl	268
ENVIRONMENTedarrayr	268
ENVIRONMENTedtabularc	268
ENVIRONMENTedtabularl	268
ENVIRONMENTedtabularr	268
ENVIRONMENTledgroup	231
ENVIRONMENTledgroupsize	231
PACKAGE(r)(e)ledmac	28
PACKAGEEledmac	10, 58, 82, 236, 289, 290, 341–343
PACKAGEEledpar	342, 343
PACKAGEEtoolbox	60
PACKAGEParallel	294
PACKAGEReledmac	293
PACKAGEamsgen	246
PACKAGEamsmath	246
PACKAGEbabel	57
PACKAGEbiblatex	54
PACKAGEbidi	61
PACKAGEccaption	67
PACKAGEcolor	56
PACKAGEedmac	1, 5, 9–12, 58, 161, 166, 213, 241, 286, 294, 333
PACKAGEedstanza	1, 12, 240
PACKAGEeledmac	1, 9, 12–14, 47, 111, 166, 232, 236, 250, 272, 283, 287, 289–291, 337, 339, 341
PACKAGEeledpar	67, 139, 270, 294, 336, 339–342
PACKAGEetoolbox	72, 111, 193, 201, 208, 225, 270, 281
PACKAGEfootmisc	27, 56, 166, 294
PACKAGEhandout	340
PACKAGEhyperref	44, 214, 237, 238, 275, 338–340
PACKAGEifluatex	60
PACKAGEifxetex	60
PACKAGEimakeidx	47, 54, 61, 232, 236, 290, 337–339, 341
PACKAGEindextool	290
PACKAGEindextools	47, 54, 61, 232, 236, 290, 341
PACKAGEinputenc	113
PACKAGEledarab	57
PACKAGEledmac	1, 9, 12, 57, 72, 236, 286, 287, 290
PACKAGEledpar	57
PACKAGEMemoir	61, 236, 290, 294, 340
PACKAGEMorewrites	55
PACKAGEMusixtex	339
PACKAGEpolyglossia	38, 57, 107, 138, 157
PACKAGERagged2e	35, 60
PACKAGEReledmac	1, 2, 9, 10, 12–14, 17, 18, 20–22, 24–28, 30, 32, 34–38, 40, 43–45, 47, 48, 53–56, 58, 59, 73, 76, 81, 82, 85, 86, 89, 97, 104, 134, 140, 143, 147, 166, 187, 194, 196, 197, 201, 208, 217, 220, 236, 250, 270, 271, 283, 290, 291, 293, 344
PACKAGEReledpar	1, 3, 5, 7, 13, 16, 37, 53, 54, 57, 59, 67, 73, 84, 89, 106, 141, 143, 182, 183, 193, 207, 208, 210, 232, 240

PACKAGESuffix .....	60
PACKAGETabmac .....	1, 12, 294
PACKAGEuninormalize .....	25
PACKAGEXargs .....	24, 60
PACKAGEXkeyval .....	59
PACKAGEXstring .....	60, 238

**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, 19
\advancepageno .....	1
\Aendnote .....	22
\affixline@num .....	1
\affixpstart@num .....	1
\affixside@note .....	1
\Afootnote .....	22
\afternoteX .....	36
\afterruleX .....	37
\ampersand .....	1, 42
\applabel .....	1, 44
\appref .....	1, 45
\apprefwithpage .....	1, 45
\arrangementX .....	1, 29
\arrangementX@normal .....	1
\arrangementX@threecol .....	1
\arrangementX@twocol .....	1
\at@every@pend .....	1
\AtEveryPend .....	1, 16
\AtEveryPstart .....	1, 16
\autopar .....	1, 15

**B**

\ballast .....	55
\ballast@count .....	1
Beeton, Barbara Ann Neuhaus Friend .....	16
\beforeeledchapter .....	1
\beforenotesX .....	36
\beginnumbering .....	1, 14
\Bendnote .....	22
\Bfootnote .....	22
\bhooknoteX .....	35

\bodyfootmarkA	27
\boxfootnotenumbers	1
Bredon, Simon	11
Breger, Herbert	11, 12, 249
Brey, Gerhard	11
Busard, Hubert L. L.	11
\bypage@false	1
\bypage@true	1
\bypstart@false	1
\bypstart@true	1

## C

\c@addcolcount	1
\c@ballast	1
\c@firstlinenum	1
\c@firstsublinenum	1
\c@labidx	1
\c@linenumincrement	1
\c@sublinenumincrement	1
\Cendnote	22
\Cfootnote	22
\ch@ck@l@ck	1
\ch@cksub@l@ck	1
\chapter	1
\check@pb@in@verse	1
Chester, Robert of	11
Claassens, Geert H. M.	11
\colalignX	35
Copernicus, Nicolaus	11
\critext	286
\ctab	1
\ctabtext	1

## D

Dekker, Dirk-Jan	56
\Dendnote	22
\Dfootnote	22
\disable@familiarnotes	1
\disable@notes	1
\disable@sidenotes	1
\disablel@dtabfeet	1
\do@actions	1
\do@actions@fixedcode	1
\do@actions@next	1
\do@ballast	1
\do@feetX	1
\do@insidelinehook	1
\do@line	1
\do@linehook	1
\do@lockoff	1

<code>\do@lockoffL</code> .....	1
<code>\do@lockon</code> .....	1
<code>\do@lockonL</code> .....	1
<code>\doedindexlabel</code> .....	1
<code>\doendnotes</code> .....	1, 22
<code>\doendnotesbysection</code> .....	1, 22
<code>\doinsidelinehook</code> .....	1, 20
<code>\dolinehook</code> .....	1, 20
<code>\dosplits</code> .....	1
Downes, Michael .....	55, 147, 148
<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, 51, 263
<code>edarrayc</code> (environment) .....	48
<code>edarrayl</code> (environment) .....	48
<code>edarrayr</code> (environment) .....	48
<code>\edatleft</code> .....	1, 50
<code>\edatright</code> .....	1, 50
<code>\edbforetab</code> .....	1, 51, 263
<code>\edfilldimen</code> .....	1
<code>\edfont@info</code> .....	1
<code>\EDINDEX</code> .....	1
<code>\edindex</code> .....	1, 46
<code>\edindexlab</code> .....	1, 48
<code>\EDLABEL</code> .....	1
<code>\edlabel</code> .....	1, 43
<code>\edlineref</code> .....	1, 43
<code>\edmakelabel</code> .....	1, 44
<code>\edpageref</code> .....	1, 43
<code>\edrowfill</code> .....	1, 49
<code>\EDTAB</code> .....	1
<code>\edtabcolsep</code> .....	1, 49
<code>\EDTABINDENT</code> .....	1
<code>\edtabindent</code> .....	1
<code>\EDTABtext</code> .....	1
<code>edtabularc</code> (environment) .....	48
<code>edtabularl</code> (environment) .....	48
<code>edtabularr</code> (environment) .....	48
<code>\EDTEXT</code> .....	1
<code>\edtext</code> .....	1, 21
<code>\edvertdots</code> .....	1, 51
<code>\edvertline</code> .....	1, 51
<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, 38
<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> .....	48
<code>edarrayl</code> .....	48
<code>edarrayr</code> .....	48
<code>edtabularc</code> .....	48
<code>edtabularl</code> .....	48
<code>edtabularr</code> .....	48
<code>ledgroup</code> .....	42
<code>ledgroupsize</code> .....	42
<code>minipage</code> .....	42
<code>Euclid</code> .....	11
<code>\extensionchars</code> .....	1, 54

## F

<code>\f@x@l@cks</code> .....	1
Fairbairns, Robin .....	27
<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, 42

<code>\flush@notes</code> .....	1
<code>\fnpos</code> .....	1, 28
Folkerts, Menso .....	11
<code>\footfootmarkA</code> .....	27
<code>\footfudgefiddle</code> .....	1, 55
<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>\footsplitskips</code> .....	1
<code>\fulllines@</code> .....	1
<code>\fullstop</code> .....	1, 38

## G

Gädeke, Nora .....	11
<code>\get@edindex@hyperref</code> .....	1
<code>\get@edindex@ledinnote@command</code> .....	1
<code>\get@index@command</code> .....	1
<code>\get@linelistfile</code> .....	1
<code>\get@sw@txt</code> .....	1
<code>\getline@num</code> .....	1
<code>\gl@p</code> .....	1

## H

<code>\h@num</code> .....	1
<code>\hangindentX</code> .....	34
<code>\hidenumbering</code> .....	1, 20
<code>\Hilfsbox</code> .....	1
<code>\hilfsbox</code> .....	1
<code>\hilfscount</code> .....	1
<code>\HILFSskip</code> .....	1
<code>\Hilfsskip</code> .....	1
<code>\hilfsskip</code> .....	1
<code>\hsizethreecolX</code> .....	35
<code>\hsizetwocolX</code> .....	35
<code>\hyperlinkformat</code> .....	1
<code>\hyperlinkformatR</code> .....	1
<code>\hyperlinkR</code> .....	1

## I

<code>\if@addsw</code> .....	1
<code>\if@edindex@fornote@true</code> .....	1
<code>\if@eled@sectioning</code> .....	1
<code>\if@led@nofoot</code> .....	1
<code>\if@lemmacommand@</code> .....	1
<code>\if@noeled@sec</code> .....	1



<code>\if@noneed@Footnote</code>	1
<code>\if@RTL</code>	1
<code>\ifautopar@pause</code>	1
<code>\ifbypage@</code>	1
<code>\ifbypage@R</code>	1
<code>\ifbypstart@</code>	1
<code>\ifbypstart@R</code>	1
<code>\ifeledmaccompat@</code>	1
<code>\iffirst@linenum@out@</code>	1
<code>\ifinserthangingsymbol</code>	1
<code>\ifistanza</code>	1
<code>\ifl@d@dash</code>	1
<code>\ifl@d@elin</code>	1
<code>\ifl@d@esl</code>	1
<code>\ifl@d@pnum</code>	1
<code>\ifl@d@ssub</code>	1
<code>\ifl@d@Xmorethantwolines</code>	1
<code>\ifl@d@Xtwolines</code>	1
<code>\ifl@dend@X</code>	1
<code>\ifl@dhidenumber</code>	1
<code>\ifl@dmemoir</code>	1
<code>\ifl@dpaging</code>	1
<code>\ifl@dpairing</code>	1
<code>\ifl@dprintingcolumns</code>	1
<code>\ifl@dprintingpages</code>	1
<code>\ifl@dskipnumber</code>	1
<code>\ifl@dskipversenumber</code>	1
<code>\ifl@dstartendok</code>	1
<code>\ifl@imakeidx</code>	1
<code>\ifl@indextools</code>	1
<code>\ifledfinal</code>	1, 54
<code>\ifledgroupnotesL@</code>	1
<code>\ifledgroupnotesR@</code>	1
<code>\iflednopbinverse</code>	1
<code>\ifledRcol</code>	1
<code>\ifledRcol@</code>	1
<code>\ifnocritical@</code>	1
<code>\ifnoend@</code>	1
<code>\ifnofamiliar@</code>	1
<code>\ifnoledgroup@</code>	1
<code>\ifnoquotation@</code>	1
<code>\ifnoteschanged@</code>	1
<code>\ifnumberedpar@</code>	1
<code>\ifnumbering</code>	1
<code>\ifnumberingR</code>	1
<code>\ifnumberline</code>	1
<code>\ifnumberstanza</code>	1
<code>\ifparapparat@</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>\insertthangingsymbol</code> .....	1
<code>\insertlines@list</code> .....	1
<code>\inserts@list</code> .....	1

## J

Jayaditya .....	11
-----------------	----

## K

Kabelschacht, Alois .....	136
---------------------------	-----

## L

<code>\l@advance@parledgroup@beforenormalnotes</code> .....	1
<code>\l@d@add</code> .....	1
<code>\l@d@nums</code> .....	1
<code>\l@d@section</code> .....	1
<code>\l@d@set</code> .....	1
<code>\l@d@Xend</code> .....	1
<code>\l@dampcount</code> .....	1
<code>\l@dbfnote</code> .....	1
<code>\l@dcheckcols</code> .....	1
<code>\l@dcheckstartend</code> .....	1
<code>\l@dchset@num</code> .....	1
<code>\l@dcolcount</code> .....	1
<code>\l@dcollect@@body</code> .....	1
<code>\l@dcollect@body</code> .....	1
<code>\l@dcolwidth</code> .....	1
<code>\l@dcsnote</code> .....	1
<code>\l@dcsnotetext</code> .....	1
<code>\l@dcsnotetext@l</code> .....	1
<code>\l@dcsnotetext@r</code> .....	1
<code>\l@ddodoreinextrafeet</code> .....	1
<code>\l@dedbeginmini</code> .....	1
<code>\l@dedendmini</code> .....	1
<code>\l@emptyd@ta</code> .....	1
<code>\l@dend@close</code> .....	1
<code>\l@dend@open</code> .....	1
<code>\l@dend@stuff</code> .....	1
<code>\l@dend@Xfalse</code> .....	1
<code>\l@dend@Xtrue</code> .....	1
<code>\l@denbody</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@dobblearg</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@dmodforedtext</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@dprintingcolumnfalse</code>	1
<code>\l@dprintingcolumntrue</code>	1
<code>\l@dprintingpagesfalse</code>	1
<code>\l@dprintingpagetrue</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@drrsn@te</code>	1
<code>\l@drrsnote</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

\l@dzero penalties	1
\label	44
\labelstartfalse	1
\labelstarttrue	1, 16
\labelref@list	1
\labelrefsparseline	1
\labelrefsparsesubline	1
\last@page@num	1
Lavagnino, John	10
\led@check@nopb	1
\led@check@pb	1
\led@err@AutoparNotNumbered	1
\led@err@edtextoutsidepstart	1
\led@err@EdtextWithoutFootnote	1
\led@err@FootnoteWithoutEdtext	1
\led@err@HighEndColumn	1
\led@err@LineationInNumbered	1
\led@err@LowStartColumn	1
\led@err@ManyLeftnotes	1
\led@err@ManyRightnotes	1
\led@err@ManySidenotes	1
\led@err@NumberingNotStarted	1
\led@err@NumberingShouldHaveStarted	1
\led@err@NumberingStarted	1
\led@err@NumberingWithoutPstart	1
\led@err@PendNoPstart	1
\led@err@PendNotNumbered	1
\led@err@PstartInPstart	1
\led@err@PstartNotNumbered	1
\led@err@ReverseColumns	1
\led@err@TooManyColumns	1
\led@err@UnequalColumns	1
\led@error@fail@patch@@doclearpage	1
\led@error@fail@patch@@iiiminipage	1
\led@error@fail@patch@@makecol	1
\led@error@fail@patch@@reinserts	1
\led@error@fail@patch@endminipage	1
\led@error@ImakeidxAfterEledmac	1
\led@error@IndextoolsAfterEledmac	1
\led@mess@NotesChanged	1
\led@mess@SectionContinued	1
\led@nopb	1
\led@nopbnum	1
\led@pb	1
\led@pb@setting	1
\led@pbnum	1
\led@toksa	1
\led@toksb	1
\led@warn@AppLabelOutEdtext	1
\led@warn@BadAction	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> .....	42
<code>ledgroupsize (environment)</code> .....	42
<code>\ledinnernote</code> .....	1, 45
<code>\ledinnote</code> .....	1
<code>\ledinnotehyperpage</code> .....	1
<code>\ledinnotemark</code> .....	1
<code>\ledleftnote</code> .....	1, 45
<code>\ledlinenum</code> .....	1
<code>\ledllfill</code> .....	1
<code>\ledlsnotefontsetup</code> .....	1, 46
<code>\ledlsnotesep</code> .....	1, 46
<code>\ledlsnotewidth</code> .....	1, 46
<code>\lednopb</code> .....	1, 53
<code>\lednopbinversetrue</code> .....	53
<code>\lednopbnum</code> .....	1
<code>\ledouternote</code> .....	45
<code>\ledouterote</code> .....	1
<code>\ledpb</code> .....	1, 53
<code>\ledpbnum</code> .....	1
<code>\ledpbsetting</code> .....	1, 53
<code>\ledrightnote</code> .....	1, 45
<code>\ledrlfill</code> .....	1
<code>\ledrsnotefontsetup</code> .....	1, 46
<code>\ledrsnotesep</code> .....	1, 46
<code>\ledrsnotewidth</code> .....	1, 46
<code>\ledsectnomark</code> .....	1
<code>\ledsectnotoc</code> .....	1
<code>\ledsetnormalparstuff@common</code> .....	1
<code>\ledsetnormalparstuffX</code> .....	1
<code>\ledsidenote</code> .....	1, 45
<code>\leftctab</code> .....	1
<code>\leftlinenum</code> .....	1, 19
<code>\leftltab</code> .....	1
<code>\leftnoteupfalse</code> .....	46

<code>\leftstartnum</code>	1
<code>\lefttab</code>	1
Leibniz	11
<code>\lemma</code>	1, 23
<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, 23
<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>\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	60

## M

<code>\m@mmf@check</code>	1
<code>\m@mmf@prepare</code>	1
<code>\M@sect</code>	1
<code>\makehboxofhboxes</code>	1
<code>\managestanza@modulo</code>	1
<code>\maxhnotesX</code>	37
Mayer, Gyula	11
<code>\measurebody</code>	1
<code>\measurecell</code>	1
<code>\measuremrow</code>	1
<code>\measuretbody</code>	1
<code>\measuretcell</code>	1
<code>\measuretrow</code>	1
Middleton, Thomas	11, 80
<code>minipage</code> (environment)	42
Mittelbach, Frank	10, 11

\morenoexpands	1, 56
\mpfnpos	1, 28
\mpnormalfootgroup	1
\mpnormalfootgroupX	1
\mpnormalvfootnote	1
\mpnormalvfootnoteX	1
\mppara@footgroupX	1
\mppara@vfootnoteX	1
\mpparafootgroup	1
\mpparavfootnote	1
\mpthreecolfootgroup	1
\mpthreecolfootgroupX	1
\mpthreecolfootsetup	1
\mpthreecolfootsetupX	1
\mptwocolfootgroup	1
\mptwocolfootgroupX	1
\mptwocolfootsetup	1
\mptwocolfootsetupX	1
\multfootsep	1, 27
\multiplefootnotemarker	1

## N

\n@num	1
\n@num@stanza	1
\new@line	1
\newhookcommand@series	1
\newhookcommand@series@reload	1
\newhooktoggle@series	1
\newhooktoggle@series@reload	1
\newseries@	1
\newverse	1
\NEXT	1
\no@expands	1
\noeledsec	53
\nomk@	1
\nonum@	1
\normal@footnotemarkX	1
\normal@page@break	1
\normal@pars	1
\normalbfnoteX	1
\normalbodyfootmarkX	1
\normalfootfmt	1
\normalfootfmtX	1
\normalfootfootmarkX	1
\normalfootgroup	1
\normalfootgroupX	1
\normalfootnoterule	1
\normalfootnoteruleX	1
\normalfootstart	1
\normalfootstartX	1

<code>\normalvfootnote</code> .....	1
<code>\normalvfootnoteX</code> .....	1
<code>\nosep@</code> .....	1
<code>\notefontsizeX</code> .....	34
<code>\notenumfontX</code> .....	34
<code>\noteschanged@false</code> .....	1
<code>\noteschanged@true</code> .....	1
<code>\noteswidthliketwocolumnsX</code> .....	37
<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> .....	17
<code>\numberlinetrue</code> .....	17
<code>\numberpstartfalse</code> .....	1, 16
<code>\numberpstarttrue</code> .....	1, 16
<code>\numberstanzafalse</code> .....	41
<code>\numberstanzatrue</code> .....	41
<code>\numlabfont</code> .....	1, 38

## O

<code>\old@hsize</code> .....	1
<code>\one@line</code> .....	1

## P

<code>\page@action</code> .....	1
<code>\page@num</code> .....	1
<code>\pagelinesep</code> .....	1, 47
<code>\pageno</code> .....	1
<code>\pageref</code> .....	44
<code>\par@line</code> .....	1
<code>\para@footgroupX</code> .....	1
<code>\para@footsetup</code> .....	1
<code>\para@footsetupX</code> .....	1
<code>\para@vfootnoteX</code> .....	1
<code>\parafootfmt</code> .....	1
<code>\parafootfmtX</code> .....	1
<code>\parafootgroup</code> .....	1
<code>\parafootsepX</code> .....	36
<code>\parafootstart</code> .....	1
<code>\parafootstartX</code> .....	1
<code>\paravfootnote</code> .....	1
<code>\parindentX</code> .....	34
<code>\pausenumbering</code> .....	1, 17
<code>\pend</code> .....	1, 15
Plato of Tivoli .....	11
<code>\postbodyfootmark</code> .....	1



<code>\prebodyfootmark</code>	1
<code>\prenotesX</code>	36
<code>\prepare@edindex@fornote</code>	1
<code>\prepare@prenotesX</code>	1
<code>\prepare@preXnotes</code>	1
<code>\prev@nopb</code>	1
<code>\prev@pb</code>	1
<code>\prevpage@num</code>	1
<code>\preXnotes</code>	1, 36
<code>\preXnotes@</code>	1
<code>\print@eledsection</code>	1
<code>\print@footnoteXrule</code>	1
<code>\print@leftmargin@eledsection</code>	1
<code>\print@line</code>	1
<code>\print@notesX</code>	1
<code>\print@rightmargin@eledsection</code>	1
<code>\print@Xfootnoterule</code>	1
<code>\print@Xnotes</code>	1
<code>\printendlines</code>	1
<code>\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>\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>	43

## Q

<code>\quotation</code>	1
<code>\quote</code>	1

## R

<code>\raggedX</code>	36
<code>\raw@text</code>	1
<code>\rbracket</code>	1, 38
<code>\read@linelist</code>	1
<code>\ref</code>	44
<code>\Relax</code>	1
<code>\reledmac@error</code>	1
<code>\reledmac@warning</code>	1
<code>\removehboxes</code>	1
<code>\resetprevline@</code>	1, 83

\resetprevpage@	1
\resetprevpage@num	83
\restore@familiarnotes	1
\restore@notes	1
\restore@sidenotes	1
\resumenumbering	1, 17
\rightctab	1
\rightlinenum	1, 19
\rightltab	1
\rightnoteupfalse	46
\rightrtab	1
\rightstartnum	1
\rigidbalance	1
\rtab	1
\rtabtext	1

## S

Sacrobosco	11
\sameword	1, 25
\sameword@inedtext	1
Schöpf, Rainer	11
\section@num	1
\select@lemmafont	1
\select@lemmafont	1, 38
\series	1
\seriesatbegin	1, 28
\seriesatend	1, 28
\set@line	1
\set@line@action	1
\setapprefprefixmore	1, 45
\setapprefprefixsingle	1, 45
\setcommand@series	1
\sethangingsymbol	1, 40
\setistwofollowinglines	1
\setl@dlp@rbox	1
\setl@drpr@box	1
\setline	1, 19
\setlinenum	1, 20
\setmcellcenter	1
\setmcellleft	1
\setmcellright	1
\setmrowcenter	1
\setmrowleft	1
\setmrowright	1
\setnoteswidthliketwocolumnsX@	1
\setnotesXpositionliketwocolumns@	1
\setprintendlines	1
\setprintlines	1
\setsidenotesep	46
\setstanzaindents	1, 39

<code>\setstanzapenalties</code>	1, 40
<code>\setstanzavalues</code>	1
<code>\settcclcenter</code>	1
<code>\settcclleft</code>	1
<code>\settcclright</code>	1
<code>\settoggle@series</code>	1
<code>\setthrowcenter</code>	1
<code>\setthrowleft</code>	1
<code>\setthrowright</code>	1
<code>\setXnotespositionliketwocolumns@</code>	1
<code>\setXnoteswidthliketwocolumns@</code>	1
<code>\showlemma</code>	1, 54
<code>\showwordrank</code>	1, 27
<code>\sidenote@margin</code>	1
<code>\sidenotemargin</code>	1, 45
<code>\sidepstartnumtrue</code>	16
<code>\skip@lockoff</code>	1
<code>\skipnumbering</code>	1, 20
<code>\splitoff</code>	1
<code>\spreadmath</code>	1, 49
<code>\spreadtext</code>	1, 49
<code>\stanza</code>	1, 39
<code>\stanza@count</code>	1
<code>\stanza@hang</code>	1
<code>\stanza@line</code>	1
<code>\stanzaindent</code>	1, 40
<code>\stanzaindent*</code>	1, 40
<code>\stanzaindentbase</code>	1, 39
<code>\stanzanumwrapper</code>	1, 41
<code>\startlock</code>	1, 19
<code>\startsub</code>	1, 19
<code>\stepl@dcclcount</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, 43
<code>\sublines@false</code>	1
<code>\sublines@true</code>	1
<code>\sublock@disp</code>	1
<code>\sublockdisp</code>	1
Sullivan, Wayne	11, 12, 55, 67, 71, 147, 148, 213, 240
<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 .....	11
<code>\thepageline</code> .....	1
<code>\thepstart</code> .....	1, 16
<code>\thestanza</code> .....	1, 41
<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 .....	11
<code>\variab</code> .....	1
<code>\vbfnoteX</code> .....	1
<code>\vl@dbfnote</code> .....	1
<code>\vl@dcnote</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> .....	33

\Xafternote	36
\Xafternumber	31
\Xafterrule	37
\Xaftersymlinenum	32
\Xarrangement	1, 29
\Xarrangement@normal	1
\Xarrangement@paragraph	1
\Xarrangement@threecol	1
\Xarrangement@twocol	1
\Xbeforelemmaseparator	33
\Xbeforenotes	36
\Xbeforenumber	31
\Xbeforesymlinenum	32
\Xhooknote	35
\Xboxlinenum	32
\Xboxlinenumalign	32
\Xboxsymlinenum	32
\Xcolalign	35
\Xdo@feet	1
\xedindex	1
\xedlabel	1
\xedtext	1
\Xendafterlemmaseparator	33
\Xendafternote	37
\Xendbeforelemmaseparator	33
\Xendbhooknote	35
\Xendboxendlinenumalign	33
\Xendboxlinenum	33
\Xendboxlinenumalign	33
\Xendboxstartlinenumalign	33
\Xendinplaceoflemmaseparator	33
\Xendinplaceofnumber	32
\Xendlemmadisablefontselection	34
\Xendlemmaseparator	33
\Xendmorethantwolines	30
\Xendmorethantwolinesapprefwithpage	45
\Xendnonumber	31
\Xendnotefontsize	34
\Xendnotenumfont	34
\Xendparagraph	37
\Xendsep	37
\Xendtwolines	30
\Xendtwolinesapprefwithpage	45
\Xendtwolinesbutnotmore	30
\Xendtwolinesbutnotmoreapprefwithpage	45
\Xendtwolinesonlyinsamepageapprefwithpage	45
\Xhangindent	34
\Xsizethreecol	35
\Xsizetwocol	35
\Xinplaceoflemmaseparator	33

\Xinplaceofnumber .....	32
\Xinsertparafootsep .....	<u>1</u>
\Xledsetnormalparstuff .....	<u>1</u>
\xleft@appenditem .....	<u>1</u>
\Xlemmadisablefontselection .....	34
\Xlemmaseparator .....	33
\xlineref .....	<u>1</u> , 43
\Xmaxhnotes .....	37
\Xmorethantwolines .....	30
\Xmorethantwolinesappref .....	45
\Xnolemmaseparator .....	<u>1</u> , 33
\Xnonbreakableafternumber .....	31
\Xnonumber .....	31
\Xnotefontsize .....	34
\Xnotenumfont .....	34
\Xnoteswidthliketwocolumns .....	37
\Xnumberonlyfirstinline .....	29
\Xnumberonlyfirstintwolines .....	30
\Xonlypstart .....	31
\xpageref .....	<u>1</u> , 43
\Xparafootsep .....	36
\Xparindent .....	34
\Xpstart .....	31
\Xpstarteverytime .....	31
\xpstartref .....	<u>1</u> , 43
\Xragged .....	36
\xright@appenditem .....	<u>1</u>
\Xstanza .....	31
\Xstanzaseparator .....	31
\xsublineref .....	<u>1</u> , 43
\Xsymlinenum .....	30
\Xtwolines .....	30
\Xtwolinesappref .....	45
\Xtwolinesbutnotmoreappref .....	45
\Xtwolinesonlyinsamepage .....	30, 45
\Xtxtbeforenotes .....	36
\Xunvxh .....	<u>1</u>
\xxref .....	<u>1</u> , 44

## Z

\zz@@@ .....	<u>1</u>
--------------	----------

## Change History

v0.1.0.	
General: First public release	1
v0.2.0.	
General: Added tabmac code, and extended indexing	1
\ifl@dmemoir: Added \ifl@dmemoir for memoir class having been used	61
\morenoexpands: Added \l@dtabnoexpands to \no@expands	104
\reledmac@error: Added \eledmac@error and replaced error messages	62
v0.2.1.	
\@lab: Removed page setting from \@lab	215
General: Added text about normal labeling	44
Bug fixes and match with mempatch v1.8	1
Major changes to insert code when memoir is loaded	211
\dextrafeet: Renamed \dextrafeet to \l@ddextrafeet	208
\edlabel: Tweaked \edlabel to get correct page numbers	213
\l@ddodoreinextrafeet: Renamed \dodoreinextrafeet to \l@ddodoreinextrafeet	209
\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	104
\zz@@@: Minor change to \zz@@@	213
v0.2.2.	
General: Improved paragraph footnotes	1
New Dekker example	1
Used \providecommand for \@gobblethree to avoid clash with the amsfonts pack- age	67
\footfudgefiddle: Added \footfudgefiddle	145
\line@list@stuff: Added initial write of page number in \line@list@stuff	98
\para@footsetup: Added \footfudgefiddle to \para@footsetup	146
\para@footsetupX: Added \footfudgefiddle to \para@footsetupX	178
v0.3.0.	
\@lab: Replaced \the\line@num by \linenumr@p\line@num in \@lab, and similar for sub-lines	215
\@nl@reg: Added a bunch of code to \@nl for handling \setlinenum	87
General: Includes edstanza and more	1
\ledlinenum: Added \linenumr@p and \sublinenum@rep to \leftlinenum and \rightlinenum	79
\linenumberlist: Added \linenumberlist mechanism	67
\printendlines: Added \linenumr@p and \sublinenumr@p to \printendlines	191
\printlines: Added \linenumr@p and \sublinenumr@p to \printlines	165
\sublinenumr@p: Added \linenumberstyle and \sublinenumberstyle	78
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	230
General: Added final/draft options	59
Added ledgroup environment	231
Added ledgroupsize environment	231
Added minipage, etc., support	1

\edtext: Added \showlemma to \edtext	105
\l@dfeetendmini: Added \l@dfeetbeginmini, \l@dfeetendmini and all their supporting code	228
\mpnormalfootgroup: Added \mpnormalfootgroup	144
\mpnormalvfootnote: Added \mpnormalvfootnote	142
\showlemma: Added \showlemma	67
\Xarrangement@normal: Added minpage footnote setup to \footnormal	141
v0.4.1.	
General: Added code for changing \@docclearpage	211
Not released. Minor editorial improvements and code tweaks	1
Only change \@footnotetext and \@footnotemark if memoir not used	167
\edindex: Let eledmac take advantage of memoir's indexing	236
\print@Xnotes: Added \@opXfeet	208
\Xdo@feet: Changed \Xdo@feet code for easier extensions	208
v0.5.0.	
\@footnotetext: Enabled regular \footnote in numbered text	167
\@xympar: Eliminated \marginpar disturbance	222
General: Added left and right side notes	222
Added sidenotes, familiar footnotes in numbered text	1
v0.5.1.	
General: Added moveable side note	222
Fixed right line numbers killed in v0.5	1
Only change \hsize in ledgroupsized environment otherwise page number can be in wrong place	231
\affixline@num: Changed \affixline@num to cater for sidenotes	129
\l@dgetsidenote@margin: Added \sidenotemargin and \sidenote@margin	222
v0.6.0.	
\@lopR: Added \@pend, \@pendR, \@lopL and \@lopR in anticipation of parallel processing	89
\@nl@reg: Added \fix@page to \@nl	87
Extended \@nl to include the page number	87
General: Fixed long paragraphs looping	1
Fixed minor typos	1
Prepared for eledpar package	1
\fix@page: Added \last@page@num and \fix@page	88
\new@line: Extended \new@line to output page numbers	98
\vl@dbfnote: Changed \l@dbfnote and \vl@dbfnote as originals could give incorrect markers in the footnotes	168
v0.7.0.	
\@nl@reg: Added \@nl@reg	87
\@ref@reg: Added \@ref@reg	95
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	61
Replaced all \interAfootnotelinepenalty, etc., by just \interfootnotelinepenalty	1
Tidying up for eledpar and ledarab packages	1
\affixline@num: Added skipnumering to \affixline@num	129
\do@actions@fixedcode: Added \do@actions@fixedcode	128



\do@actions@next: Added number skipping to \do@actions	126
\do@insidelinehook: Added \do@linehook for use in \do@line	124
\endnumbering: Changed \endnumbering for eledpar	70
\f@x@l@cks: Added \ch@cksub@l@ck, \ch@ck@l@ck and \f@x@l@cks	131
\footsplitskips: Added \footsplitskips for use in many footnote styles	139
\get@linelistfile: Added \get@linelistfile	85
\ifledRcol@: Added \l@dunpstartsL, \ifl@dpairing and \ifpst@rted for/from eledpar	67
\initnumbering@reg: Added \initnumbering@reg	69
\l@advance@parledgroup@beforenormalnotes: Added \l@dunboxmpfoot containing some common code	230
\l@dcsnotetext@r: Added \l@demptyd@ta	125
\l@dgetline@margin: Added \l@dgetline@margin	75
\l@dgetlock@disp: Added \l@dgetlock@disp	77
\l@dgetsidenote@margin: Added \l@dgetsidenote@margin	222
\l@drsn@te: Added \l@dlsn@te and \l@drsn@te for use in \do@line	125
\l@dzeropenalties: Added \l@dzeropenalties	120
\ledlinenum: Added \ledlinenum for use by \leftlinenum and \rightlinenum	79
\line@list@stuff: Deleted \page@start from \line@list@stuff	98
\list@clearing@reg: Added \list@clearing@reg	85
\n@num: Added \n@num	94
\normalbfnoteX: Removed extraneous space from \normalbfnoteX	172
\resumenumbering: Changed \resumenumbering for eledpar	71
\setprintendlines: Added \setprintendlines for use by \printendlines	189
\setprintlines: Added \setprintlines for use by \printlines	162
\skipnumbering: Added \skipnumbering and supports	101
\sublinenumincrement: Added \firstlinenum, \linenumincrement, \firstsublinenum and \linenumincrement	76
\sublinenumr@p: Using \linenumrep instead of \linenumr@p	78
Using \sublinenumrep instead of \sublinenumr@p	78
\vnumfootnoteX: Removed extraneous space from \vnumfootnoteX	173
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 . . . . .	67
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. . . . .	39
New stanzaindentsrepetition counter: to repeat stanza indents every $n$ verses. . . . .	1
\managestanza@modulo: New stanzaindentsrepetition counter to repeat stanza indents every $n$ verses. . . . .	242
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. . . . .	213
v0.15.0.	
General: Line numbering can be reset at each pstart. . . . .	73
Possibility to print \pstart number inside. . . . .	16
\affixline@num: Line numbering can be disabled. . . . .	129
\ifinserthangingsymbol: New management of hangingsymbol insertion, preventing undesirable insertions. . . . .	240
\printlines: Line numbering can be reset at each pstart. . . . .	165
v0.17.0.	
\ifinserthangingsymbol: New new management of hangingsymbol insertion, preventing undesirable insertions. . . . .	240
v1.0.0.	
General: \lemma can contain commands. . . . .	23
Debug in lineation command . . . . .	18
New generic commands to customize footnote display. . . . .	28, 201
Options nonum and nosepl in \Xfootnote. . . . .	22
Options of \Xfootnotes. . . . .	137
Possibility to have commands in sidenotes. . . . .	45
Some compatibility break with eledmac. Change of name: eledmac. . . . .	1
\morenoexpands: Change to be compatible with new features . . . . .	104
v1.0.1.	
General: Correction on \Xnumberonlyfirstinline with lineation by pstart or by page. . . . .	29
v1.1.0.	
General: Add \labelpstarttrue. . . . .	16
Add \Xnumberonlyfirstintwolines . . . . .	30
Add \Xpstart and \Xonlypstart . . . . .	31
New hook to add arbitrary code at the beginning of the notes . . . . .	35
New options for block of notes. . . . .	36
New package option: parapparatus. . . . .	1
New tools to change order of series . . . . .	200
Sectioning commands. . . . .	51
\preXnotes: New skip \preXnotes@ . . . . .	185
\settoggle@series: \settoggle@series switch the global value of the toggle, not only the local value. . . . .	202

v1.2.0.	
\endquote:	Compatibility of \ledchapter with the <i>memoir</i> class. . . . . 268
\preXnotes:	Debug in familiar footnotes (bug introduced by v1.1). . . . . 185
v1.3.0.	
\endquote:	<i>Quotation</i> and quote environment inside numbered sections. . . . . 268
v1.4.0.	
General:	Compatibility with LuaTeX of RTL notes. . . . . 1
\edtext:	Compatibility of \edtext with the right-to-left direction (with Polyglossia). 105
\ledsetnormalparstuffX:	Direction of footnotes with polyglossia. . . . . 182
\newseries@:	Remembers the language of the lemma, in order to create a correct direc- tion for the footnote separator. . . . . 195
\rbracket:	Switch the right bracket to a left bracket when the lemma is RTL (needs polyglossia or LuaTeX). . . . . 157
v1.4.1.	
\affixside@note:	Remove spurious spaces. . . . . 227
\endquote:	New option <i>noquotation</i> . . . . . 268
\labelrefsparsesubline:	Fix bug with \edlabel. . . . . 214
\vl@dbfnote:	Compatibility of standard footnotes with eledmac when these footnotes contain any commands. . . . . . 168
v1.4.2.	
General:	Debug with some special classes. . . . . 1
v1.4.3.	
General:	Add \Xnonbreakableafternumber. . . . . 31
	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. . . . . 268
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. . . . . 172
v1.5.0.	
General:	Correct indexing when the call is made in critical notes. . . . . 232
\do@insidelinehook:	Added \do@insidelinehook for use in \do@line . . . . . 124
\edindex:	Compatibility with imakeidx package, and possibility to use multiple index with \edindex. . . . . 236
v1.5.1.	
\managestanza@modulo:	Correct stanzaindentsrepetition counter . . . . . 242
\normalvfootnoteX:	Fix bug with normal familiar footnotes when mixing RTL and LTR text. . . . . 169
v1.6.0.	
\newverse:	Add \falseverse macro. . . . . 244
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. . . . .	117
\ifinserthangingsymbol: Hang verse is now not automatically flush right. . . . .	240
\l@dunhbox@line: Move the call to \inserthangingsymbol to allow use \hfill in-side. . . . .	122
\pend: Spurious space in \pend. . . . .	119
v1.7.0.	
General: New features for managing page breaks. . . . .	53
v1.8.0.	
General: Compatibility with parledgroup option of eledpar package. . . . .	1
If imakeidx and hyperref are loaded, adds hyperref in the index. . . . .	232
\endquote: Correction of sectioning commands in parallel texts. . . . .	268
\get@index@command: Debug \get@index@command and compatibility with hyperref package. . . . .	235
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work. . . . .	204
\prevpage@num: Correct \parafootsep when using with ledgroup. . . . .	151
v1.8.1.	
General: Debug endnotes when more than one series is used (change the position where tools for endnotes are defined). . . . .	186
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. . . . .	232
v1.9.0.	
\doxtrafeet: Add \fnpos to choice the order of footnotes. . . . .	208
\l@dfeetendmini: Add \mpfnpos to choice the order of footnotes in minipage / led-group. . . . .	228
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. . . . .	268
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. . . . .	34
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. ....	86
General: Add \ledinnernote and \ledouternote commands. ....	45
Add \Xendparagraph and related settings. ....	37
Add hyperlink to crossref (needs hyperref package). ....	42
Compatibility with musixtex. ....	1
Debug eledmac sectioning command after using \resumenumbering. ....	1
Ensure that imakeidx is loaded <i>before</i> eledmac ....	232
New hooks: \Xafterrule and \afterruleX ....	37
New options for ragged-paragraph notes ....	36
New sectioning commands. ....	51
Optional arguments for \pstart and \pend. ....	16
\AtEveryPstart: New optional argument for \pstart, to execute code before it. ...	117
\edindex: Use correctly default index when imakeidx is loaded. ....	236
\endquote: \ledxxx sectioning commands are deprecated and replaced by \eledxxx commands. ....	268
\ifledRcol@: Add \ifledRcol@ for eledpar ....	67
\initnumbering@reg: \beginnumbering is defined only on eledmac, not on eledpar. ....	69
\l@dcnote: \l@dlsnote, \l@drsnote and \l@dcnote defined only one time, in eledmac, including needs for eledpar case. ....	224
\l@dgetsidenote@margin: \sidenotemargin is now directly defined in eledmac to be able to manage eledpar. ....	222
\l@dunhbox@line: \do@line is split in more little commands. ....	122
\newhookcommand@series@reload: Debug \beforenotesX and \maxhnotesX which did not work when called after \footparagraphX. ....	204
Debug \Xbeforenotes and \Xmaxhnotes which did not work when called after \footparagraph. ....	204
\pend: New optional argument for \pend, to execute code after it. ....	119
\stanza: &can have an optional argument: content to be printed after. ....	244
\Stanza can have an optional argument: content to be printed before. ....	244
Add \newverse macro, \falseverse deprecated. ....	244

## v1.12.1.

\wrap@edcrossref: Fix spurious spaces. ....	217
---	-----

## v1.12.2.

\l@dunhbox@line: Fix a bug with critical notes at the tops of pages (added by v12.0.0) ...	122
--	-----

## v1.12.3.

General: Add macros for new messages since v0.7 ....	61
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. ....	45
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 ....	99
\flag@start send a error message when a \edtext is done without insert (note) ...	99
\reledmac@error: Replaced error messages ....	62

## 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	237
\@ssect: Debug \eledchapter in parallel with memoir	272
\doinsidelinehook: Added \dolinehook and \doinsidelinehook	125
\endnumbering: Allow to mix parallel columns and normal text when using \pausenumbering	70
\l@dgobblearg: \l@dgobblearg becomes \l@gobbeloptarg	252
\l@drestoreforedtext: Debug optional arguments of \Xfootnote in tabular context	252
\resumenumbering: Debug \resumenumbering	71
v1.12.7.	
\wrap@edcrossref: \wrap@edcrossref is now robust	217
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	99
v1.13.0.	
General: Add \Xnoteswidthliketwocolumns and \noteswidthliketwocolumnsX	37
Added widthliketwocolumns option	59
\newhooktoggle@series@reload: Add \newhookcommand@toggle@reload	204
\para@footsetupX: In \para@footsetupX, use \columnwidth instead of \hsize	178
\settoggle@series: \settoggle@series can take an optional arguments to reload series setup.	202
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	117
v1.13.2.	
General: Fix bug with normal footnotes, added by v1.13.0.	1
\ifledRcol@: Add \ifl@dpadding for eledpar	67
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.	186
Delete previous override of \l@d@@wrindexhyp at the beginning of a document when hyperref is not loaded.	239
Move gobbling command	67
Provide \@gobblefour	67
\edindex: Let eledmac take advantage of imakeidx even when memoir class is used	236
v1.14.1.	
\@ssect: Debug sectioning commands when using both handout and hyperref package.	275
v1.14.2.	
\@ssect: Debug \edtext after starred sectioning commands when using memoir class.	272
v1.15.0.	
\@edtext@level: New boolean \if@edtext@.	105
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 . . . . .	24
\arrangementX@threecol: Correct bug with paragraphed familiar footnotes setting. . . . .	178
\endsub: Restore subline feature (disabled by mistake in v1.8.0). . . . .	100
\if@lemmacommand@: New boolean \iflemmacommand@. . . . .	110
v1.15.1.	
\line@list@stuff: Revert modification of 1.5.2 which makes bug with numbering. Leave vertical mode to solve spurious space before minipage. . . . .	98
v1.16.0.	
General: \edtext is now defined only in eledmac, not in eledpar. Debug wrong num- bering when using \sameword + eledpar + \tag command. . . . .	105
Compatibility of standard footnotes with some biblatex styles. . . . .	1
New \stanzaindent command. . . . .	1
v1.16.1.	
\edlineref: \lineref is not defined if defined by some other package, like lineno. Eledmac provides \edlineref instead. . . . .	217
v1.17.0.	
\edtext: Error message when calling \edtext outside of a numbered paragraph. . . . .	105
v1.18.0.	
\@edindex@hyperref: Fix spurious space with \edindex when using imakeidx/indextools + hyperref. . . . .	237
General: Add \Xpstarteverytime . . . . .	31
Compatibility with Lua $\TeX$ RTL languages. . . . .	1
Debug \Xonlypstart when using \Xnumberonlyfirstinline and the current line number differs from the previous. . . . .	31
\edlabel: \edlabel is now defined only one time for both eledmac and eledpar . . . . .	213
\ifledRcol@: Add \ifl@dprintingpages and \@dprintingcolumns for eledpar . . . . .	67
\l@d@section: Option parapparatus works for endnotes. . . . .	187
\print@line: Compatibility with Lua $\TeX$ RTL languages. . . . .	123
\printlinefootnote: Code refactoring in \printlinefootnote: the printing of the numbers are factorized in \printlinefootnotearea . . . . .	158
\printpstart: Debug \Xpstart with parallel pages and columns (eledpar) . . . . .	158
v1.19.0.	
General: \Xmaxhnotes and \maxhnotesX work now for both two-columns and three- columns setting. . . . .	1
Compatibility with eledpar v1.13.0. . . . .	1
\footplitskips: \footplitskips doesn't set \floatingpenalty to \@MM when processing parallel pages. . . . .	139
\xxref: \xxref works also with right side numbers, when \@Rlineflag is not empty. . . . .	219
v1.19.1.	
General: Call \correct@footinsX@box and \correct@Xfootins@box directly in \print@notesX@forpages and \print@Xnotes@forpages, that is in eledpar. . . . .	1
v1.20.0.	
General: Add \Xendboxlinenum . . . . .	33
Add \Xtwolines and \Xmorethantwolines hooks . . . . .	30
Add series option. . . . .	1
Correct \Xinplaceofnumber hook. . . . .	1
Explicit error message when calling \Xfootnote outside of \edtext. . . . .	1
Fix bug with line number typesetting direction when using \eledsection and similar commands for RTL texts with Lua $\TeX$ . . . . .	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>	165
<code>\stanza</code> : <code>&amp;</code> and <code>\&amp;</code> can be preceded by spaces.	244
<code>\xxref</code> : Debug <code>\xxref</code> when not loading <code>eledpar</code> (fix bug added in 1.19.0).	219
v1.21.0.	
<code>\@edindex@hyperref</code> : Look at the hyperindex option of <code>hyperref</code> before inserting <code>hyperref</code>	237
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	200
Add <code>\applabel</code> and related	44
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	35
Add <code>\Xendtwolines</code> , <code>\Xendmorethantwolines</code> , <code>\Xendtwolinesbutnotmore</code> and <code>\Xendtwolinesonlyinsamepage</code> .	30
Add <code>\Xparindent</code> and <code>\hangindentX</code>	34
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>	67
<code>\l@d@section</code> : <code>\endnotes</code> take five arguments.	187
<code>\ledinnotemark</code> : Add <code>\ledinnotemark</code> .	235
<code>\ledsetnormalparstuffX</code> : <code>\ledsetnormalparstuff</code> is deprecated and becomes <code>\ledsetnormalparstuffX</code> and <code>\Xledsetnormalparstuff</code> .	182
<code>\n@num</code> : <code>\n@num@ref</code> deleted	94
<code>\n@num</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code>	94
<code>\newhookcommand@series</code> : <code>\newhookcommand@series</code> can take an optional argument.	203
<code>\newhooktoggle@series</code> : <code>\newhooktoggle@series</code> can take an optional argument.	204
<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>	184



<code>\seriesatend</code> : Fix spurious space in <code>\seriesatend</code> . . . . .	200
<code>\skipnumbering</code> : <code>\skipnumbering</code> defined only one time for both <code>Eledmac</code> and <code>Eledpar</code> . . . . . .	101
Correct <code>\skipnumbering</code> for stanza. . . . .	101
Delete <code>\skipnumbering@reg</code> . . . . .	101
v1.22.0.	
General: Add <code>\doendnotesbysection</code> command. . . . .	22
Add option for lemma separator inside endnotes . . . . .	33
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. . . . .	22
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. . . . .	235
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>\Xsymllinenum</code> . . . . . .	151
v1.23.0.	
<code>\@edtext@level</code> : The boolean <code>\if@edtext@</code> becomes the counter <code>\edtext@level</code> . . . . .	105
General: Add <code>\Xboxlinenumalign</code> and <code>\Xendboxlinenumalign</code> . . . . .	32
Add <code>\Xboxstartlinenum</code> , <code>\Xendboxstartlinenum</code> , <code>\Xboxendlinenum</code> , <code>\Xendboxendlinenum</code> . . . . . .	32
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. . . . .	24
Fix bug with <code>\sameword</code> when the same lemma is used for multiple notes or for nested <code>\edtexts</code> . . . . .	24
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. . . . .	24
<code>\lemma</code> : Fix spurious space after <code>\lemma</code> command . . . . .	109
<code>\newseries@</code> : Prevent spurious spaces when <code>\Afootnote</code> and similar commands are followed by spaces (bug added on 1.0.0). . . . .	195
<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. . . . .	113
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: \lemma 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 \edindex when using xindy+hyperref option.	1
v1.24.5.	
General: Fix bug of indent, when a added in 1.1.0, when a \beginnumbering immediately follow a sectioning command.	1
v2.0.0.	
\@iiiminipage: Patch \@iiiminipage instead of redefining it.	230
\@xympar: Patching \@xympar instead of redefining it	222
General: \@makecol, \@reinserts and \@docclearpage are patched instead of begin redefined	211
\doxtrafeeti becomes \do@feetX; \doxtrafeetii becomes \Xdo@feet; \@opxtrafeeti becomes \@opfeetX; \doreinxtrafeetii becomes \X@doreinfeet; \doreinxtrafeeti becomes \@doreinfeetX.	211
Add \Xendinplaceofnumber hook.	1
Add \Xendnonumber hook.	1
Add nonum option for endnotes.	1
Fix bug when printing only one series of endnotes, but wanted to keep endnotes for other series.	1
In order to have a more consistent name’s convention, many names has been changed.	1
Many L <sup>A</sup> T <sub>E</sub> X’s output macros are now patched and not override.	1
Package’s name becomes reledmac.	1
Patch \@footnotemark instead of redefine it	167
Suppress indexing command specific to memoir.	236
\endminipage: Patch \endminipage instead of redefining it.	230
\initnumbering@quote: \initnumbering@sectcmd becomes \initnumbering@quote	268
\l@advance@parledgroup@beforenormalnotes: Some conde of \l@dumboxmpfoot moved to \l@advance@parledegroupp@beforenormalnotes	230
\newseries@: One endnotes file by series.	198
v2.0.1.	
General: Fix bug in eledmac-compat option	1
Fix incompatibility between optional argument of \pstart and \numberpstarttrue	1
v2.1.0.	
General: Fix bug with \advanceline at the begin of a \pstart.	1
Fix bug with \chapter in optional argument of \pstart in parallel typesetting with scrbook.	1
Fix bug with \eledchapter in parallel typesetting with scrbook.	1
Fix bug with \setline at the begin of a \pstart.	1
Fix space bug with \Xbhooknote and \bhooknoteX when using to insert text and not to execute code.	1
New tools to number stanza	1
v2.1.1.	
General: Fix bug with \ledpbsetting{before}.	1

v2.1.2.

General: Fix bug with lineation by pstart and tabular environment (added in 2.1.0). . . . . 1